IBM

# S/390 Partners in Development: ThinkPad Enabled for S/390

- System setup
- AD CD-ROM use
- System operation

Bill Ogden

Redbooks

ibm.com/redbooks

International Technical Support Organization

# S/390 Partners in Development: ThinkPad Enabled for S/390

October 2001

SG24-6507-00

**Take Note!** Before using this information and the product it supports, be sure to read the general information in “Special notices” on page 85.

**First Edition (October 2001)**

This edition applies to the OS/390 AD CD-ROM and FLEX-ES versions available at the time this book was published.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HYJ  Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

**© Copyright International Business Machines Corporation 2001.  All rights reserved.**
Note to U.S Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

# Contents

**Contents** . . . . . iii

**Preface** . . . . . vii
The author . . . . . vii
Special notice . . . . . vii
IBM trademarks . . . . . viii
Comments welcome . . . . . viii

**Chapter 1. Introduction** . . . . . 1
1.1 Purpose of this redbook . . . . . 2
1.2 ThinkPad/EFS systems . . . . . 2
1.2.1 FLEX-ES . . . . . 3
1.2.2 Linux . . . . . 4
1.2.3 Positioning with other small S/390s . . . . . 4
1.3 ThinkPad/EFS hardware used . . . . . 5
1.4 Terminology . . . . . 6

**Chapter 2. System and Linux installation** . . . . . 7
2.1 Disk planning . . . . . 7
2.2 ThinkPad . . . . . 9
2.3 Linux installation . . . . . 9
2.3.1 Installation notes . . . . . 12

**Chapter 3. FLEX-ES and OS/390 installation** . . . . . 15
3.1 FLEX-ES . . . . . 16
3.1.1 Brief introduction . . . . . 16
3.1.2 Installation . . . . . 19
3.1.3 X3270 . . . . . 20
3.1.4 Next steps . . . . . 21
3.2 OS/390 AD systems . . . . . 21
3.2.1 AD systems . . . . . 21
3.2.2 OS/390 on CD-ROM . . . . . 22
3.2.3 OS/390 device configuration . . . . . 23
3.2.4 An unzip program . . . . . 24
3.2.5 Installation tasks . . . . . 24
3.3 File ownership . . . . . 25
3.4 Minor problems . . . . . 25
3.4.1 Next steps . . . . . 26

**Chapter 4. FLEX-ES Operation** . . . . . 27
4.1 FLEX-ES system and resource definitions . . . . . 28
4.1.1 Building a shell script . . . . . 29
4.1.2 IPL OS/390 . . . . . 31
4.2 Operation and use . . . . . 32
4.2.1 User terminal connection . . . . . 32
4.2.2 Linux TN3270 . . . . . 33
4.2.3 IODF requirements . . . . . 33
4.2.4 System performance monitors . . . . . 33
4.2.5 TCP/IP for OS/390 . . . . . 34

© Copyright IBM Corp. 2001 

4.3 Shutting down . . . . . 35

**Chapter 5. Additional Topics** . . . . . 37
5.1 Basic debugging . . . . . 38
5.2 Operating Systems Messages console . . . . . 39
5.3 Security . . . . . 39
5.4 Server memory . . . . . 39
5.4.1 The vmstat command . . . . . 40
5.4.2 Importance of Linux swapping . . . . . 41
5.5 Ultrabay . . . . . 42
5.5.1 Using a second Linux hard disk . . . . . 42
5.5.2 Second disk planning . . . . . 43
5.5.3 Disk layout (AD system) . . . . . 44
5.5.4 Alternative method . . . . . 45
5.6 FLEX-ES FakeTape on OS/390 . . . . . 45
5.7 Multi-system setup . . . . . 47
5.8 x3270 client . . . . . 48
5.9 Remote resources . . . . . 49
5.10 Disk caches . . . . . 50
5.11 Tuning cachesize . . . . . 52
5.12 FSI Channel Adapters . . . . . 52
5.13 SCSI adapter for the ThinkPad . . . . . 53
5.14 Disk fragmentation . . . . . 53
5.15 Backup and restore considerations . . . . . 53
5.15.1 Using tar to back up S/390 volumes . . . . . 54
5.15.2 Using tar and ftp . . . . . 55
5.15.3 CD-RW drive . . . . . 56
5.16 S/390 identification . . . . . 56
5.17 Display PSW and registers . . . . . 57
5.18 RAS discussion . . . . . 57
5.19 Verify ckd disk . . . . . 58
5.20 Linux windows . . . . . 58
5.21 *Installing FLEX-ES upgrades* . . . . . *59*
5.22 ThinkPad power control . . . . . 59
5.23 Useful Linux commands . . . . . 60

**Chapter 6. Frequently asked questions** . . . . . 63

**Appendix A. FLEX-ES definition listings** . . . . . 69
Basic definitions for a single HDD . . . . . 69
Shell script for a single HDD . . . . . 70
Definitions for two HDDs . . . . . 71
Shell script for two HDDs . . . . . 72

**Appendix B. FLEX-ES parameters** . . . . . 73
System definitions . . . . . 73
Resource definitions . . . . . 75
Emulated control unit types . . . . . 75
Emulated device types . . . . . 76
Typical resource definitions . . . . . 76
Cloned devices . . . . . 80
Common rules . . . . . 81
The resadm command . . . . . 81
CLI commands . . . . . 82

**Special notices** . . . . . 85

**Related publications** . . . . . 87
IBM Redbooks . . . . . 87
Other resources . . . . . 87
Referenced Web sites . . . . . 87
How to get IBM Redbooks . . . . . 87
IBM Redbooks collections . . . . . 88

**Index** . . . . . 89

# Preface

A ThinkPad Enabled for S/390, generally known as a ThinkPad/EFS system, is the smallest S/390 currently available, and is intended for development and demonstration purposes.

It is based on an IBM ThinkPad running Linux, and the S/390 emulation product FLEX-ES. FLEX-ES is a product of Fundamental Software, Incorporated (FSI) of Fremont, California. This package (the ThinkPad, Linux, and FLEX-ES) can run all current S/390 operating systems.

This IBM Redbook introduces the ThinkPad/EFS system, describes the setup process of the system in some detail, and then describes the installation and use of an OS/390 package known as the OS/390 AD CD-ROM system.

While this publication is primarily directed at members of the IBM S/390 Partners in Development program (also known as PartnerWorld), most of the content applies to any ThinkPad/EFS system.

## The author

**Bill Ogden** is a retired IBM Senior Technical Staff Member, still working part-time on favorite projects at the International Technical Support Organization, Poughkeepsie Center. He specializes in entry-level OS/390 and z/OS systems, writes extensively, and teaches ITSO workshops relating to these areas. Bill has been with the ITSO since 1978. Many of his projects for the last several years have been related to IBM's S/390 Partners in Development organization.

Thanks to the following people for their contributions to this project:

**Jim Obrizok**, **Linda Lovallo**, and **Frank Yolton**, of the S/390 Partners in Development program, who coordinated the PID elements of the project and helped obtain required hardware.

**Scott Carter, Martin Groen, Peter Ward, David MacMillan, Ray La Croix,** and **Tom Valerio,** of Fundamental Software, Inc., who provided considerable help with the details of planning and installation.

**Mike Hammock**, of IntelliWare Systems, Inc., who provided useful feedback about typical system use.

## Special notice

This publication is intended to help new users of ThinkPad/EFS systems better understand the platform. The information in this publication is not intended as the specification of any programming interfaces. See the PUBLICATIONS section of any relevant IBM Programming Announcement for more information about what publications are considered to be product documentation.

© Copyright IBM Corp. 2001

# IBM trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

e (logo)®
IBM ®
AIX®
CICS®
DB2®
ESCON®
IMS™
Multiprise®
MVS™
Netfinity®
NUMA-Q®
OS/2®
OS/390®
PartnerWorld®
Perform™
PS/2®
RAA®
Redbooks™
Redbooks Logo
RACF®
RMF™
S/370™
S/390®
SP™
ThinkPad®
Ultrabay™
VM/ESA®
VSE/ESA™
VTAM®
WebSphere®
xSeries™
z/OS™
z/VM™

# Comments welcome

Your comments are important to us!

We want our IBM Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  **ibm.com**/redbooks

- Send your comments in an Internet note to:

  redbook@us.ibm.com

- Mail your comments to the address on page ii.

1

# Introduction

Netfinity/EFS and ThinkPad/EFS systems are entry-level S/390 products. Implementation, involving underlying PC Server (or ThinkPad) hardware and S/390 emulation software, differs considerably from earlier machines. The EFS portion of the name is derived from “Enabled For S/390.”

EFS systems may be used to run OS/390 (and z/OS[1]), VM/ESA (and z/VM), VSE/ESA, and Linux for S/390. They may also be used for older versions of these operating systems. This document addresses only OS/390 and z/OS (31-bit mode) use, but the reader should not infer that the EFS is limited to OS/390 and z/OS.

Rather than constantly write “OS/390 and z/OS (31-bit mode)” we tend to simply mention OS/390 unless there is a particular point to be made about z/OS. You should assume that this book refers to both OS/390 and z/OS.

[1] z/OS (31-bit mode) is supported in the latest release for FLEX-ES. Much of the work for this redbook was done before this latest release was available. We expect no significant differences to the redbook contents if you are using z/OS.

© Copyright IBM Corp. 2001

## 1.1 Purpose of this redbook

This IBM Redbook introduces ThinkPad/EFS systems and describes the installation process for a particular configuration of these systems. The configuration used is the base configuration selected by IBM's S/390 Partners in Development (PID)[2] program for members of that program. Many of the specific details described here are for these PID systems and the OS/390 (and z/OS) software packages IBM provides for PID members who obtain systems through this program. However, the descriptions in this book can be generalized to apply to most other ThinkPad/EFS systems and installations.

ThinkPad/EFS systems are available only through IBM business partners, and are not available directly from IBM. In normal situations, the business partner performs the initial machine setup—including hardware installation, basic Linux operating system installation, initial disk configuration, and FLEX-ES installation. This redbook describes these same steps and, in a sense, describes steps that a normal ThinkPad/EFS user should not need to perform.

We include this material for several reasons:

- Understanding how a system is set up, even though someone else may have done it for you, leads to a better understanding of the system.
- The system owner may accidently (or intentionally) delete or destroy part of the underlying elements of his system. If this happens, he might ask his IBM business partner for assistance, or he might prefer to rebuild the system himself.
- Significant upgrades to the underlying elements (hardware, Linux, FLEX-ES) may require a partial or complete rebuilding of the system. Again, the system owner has the option of using business partner assistance or implementing the upgrade himself.

After describing initial system setup, we describe the installation of an OS/390 Application Development CD-ROM system. These are usually known as AD systems, and are available only to PID members who obtain systems through the PID program. The OS/390 (and z/OS) systems involved are typical small implementations of these operating systems. The only unique factor is the packaging on CD-ROM. If you are not a PID member, this section of the document may not be of interest.

Another redbook, *Netfinity Enabled for S/390*, SG24-6501, duplicates much of the material in this *ThinkPad Enabled for S/390* book. The ThinkPad material is based on Linux, while the Netfinity material is based on UnixWare or Open UNIX.[3] A third redbook, *NUMA-Q Enabled for S/390*, SG24-6215, is based on different text and is a recommended companion to both the Netfinity and ThinkPad books, even though the NUMA-Q EFS system is no longer available.

## 1.2 ThinkPad/EFS systems

A ThinkPad/EFS machine is an IBM ThinkPad laptop running Linux (the Red Hat 7.1 distribution for the discussions in this redbook). A Linux application package, FLEX-ES, emulates a S/390 (processor and I/O). The emulated S/390 is used to run OS/390 and all the normal subsystems and applications that run under OS/390. FLEX-ES emulates typical S/390 I/O devices such as 3390 disks, 3480/3490 tape drives, and 3174 controllers. Normal 3270 terminal connections are through TN3270 sessions to a FLEX-ES program that makes these sessions appear to the emulated S/390 as local, non-SNA 3270 devices.

---

[2] Also known as IBM's PartnerWorld organization.

[3] Open UNIX 8.0 is the replacement and upgrade product for UnixWare 7.1.1. References in this document to UnixWare also apply to Open UNIX.

No “real” S/390 hardware is required, although there are options to attach parallel and ESCON channel devices.[4] No special adapter cards or other hardware is needed for basic S/390 emulation. The emulated S/390 runs as a Linux process, in the virtual memory provided by Linux. Every S/390 instruction that is executed is interpreted by the FLEX-ES emulation program, which then uses PC instructions to emulate the effects of the S/390 instruction.[5]

ThinkPad/EFS is not intended as a production S/390 machine. Many reasons are involved, including obvious ones such as:

- No RAID protection is available for emulated S/390 disk volumes.
- Total disk space is limited, as is the configuration of the disk space.
- No S/390 channel connectivity is supported.
- Memory is limited and must be shared between S/390 use and all the Linux-based processes.

The system is intended as a small development machine, a demonstration machine, and may fit in a number of debugging situations. It is definitely a multi-user system, with, for example, TSO users connected through TN3270 sessions.

### 1.2.1 FLEX-ES

FLEX-ES is a product of Fundamental Software, Incorporated, (FSI) of Fremont, California. ThinkPad/EFS systems are a cooperative effort involving IBM, FSI, and several business partners that market ThinkPad/EFS systems. FLEX-ES runs under Linux or several UNIX operating systems.

FSI also produces four hardware options that may be used with FLEX-ES. These are:

- A parallel channel adapter, providing one S/390 parallel channel
- A parallel channel adapter, providing three S/390 parallel channels
- An ESCON channel adapter, providing one S/390 ESCON channel[6]
- A communications adapter, providing six lines corresponding to S/370 ICA lines

These adapters are full-length PCI adapters and would not normally be used with a laptop machine. In principle, it may be possible to use one by installing it in an appropriate ThinkPad docking station. At the time of writing, such options had not been tested and must be considered unsupported; furthermore, the required FLEX-ES support modules are not provided with ThinkPad/EFS systems. This redbook ignores the FSI hardware adapters. If you are interested in these, you should refer to the NetFinity/EFS redbook.

FLEX-ES is a licensed product. The licenses are arranged through IBM and FSI business partners. A FLEX-ES license specifies the number of PC processors that can be used for S/390 emulation. For the ThinkPad/EFS system described in this redbook, this will specify a single processor. A FLEX-ES license is keyed to a specific serial number in a hardware *dongle* that is connected to the USB port of the ThinkPad and will run only in a ThinkPad that has this dongle installed. It is provided as part of the FLEX-ES license. A FLEX-ES license is also keyed to a specific PC Server (ThinkPad) speed, and will operate only on a server that operates within 2% of the specified speed.

---

[4] At the time of writing, there is no practical support for these adapters on a ThinkPad. We mention them here for completeness.

[5] This is a very brief statement of a complex process. FLEX-ES internally uses a “just in time” compiler technique to build PC code that emulates the S/390 instructions. A controlled amount of the “compiled” code is retained to improve performance within loops and so forth.

[6] This adapter was not yet available at the time of writing.

## 1.2.2 Linux

In order to provide well-tested S/390 operation, a single Linux distribution was chosen as the ThinkPad base software. This is Red Hat 7.1 Linux. Other versions and distributions of Linux might work, but only ThinkPad/EFS machines running on this base operating system are supported at the time of writing.

This may change in the future, of course. Linux is an ongoing process and the EFS base system is likely to migrate to a future Linux kernel when one becomes available and FSI migrates the FLEX-ES code to it. FSI and IBM intend to manage EFS development with a high priority for system stability. In general, this means the EFS base operating system will not "chase" small Linux enhancements and may delay movement to major new releases until their stability *for the EFS environment* has been thoroughly confirmed.

Linux is a large topic. This redbook addresses Linux only to the extent needed to install and use FLEX-ES. There are many clever things that can be done with Linux, and some of these can interact with FLEX-ES and files used with FLEX-ES. Except as needed to install a basic ThinkPad/EFS system, we do not explore the things a user might implement using Linux facilities.

## 1.2.3 Positioning with other small S/390s

Recent entry-level S/390 machines used by PID members have included the following:

- P/390, R/390, and S/390 Integrated Server (IS) systems (all based on P/390 adapters). These systems are no longer manufactured or marketed, but are still widely used by members of IBM's Partners in Development (PID) organization.
- Application StarterPak systems. These are no longer manufactured or marketed, but a number are in use by PID members.
- Several models of Multiprise 3000 systems (MP3000). These are currently manufactured and marketed and widely used.
- x/Server EFS (formerly known as NUMA-Q Enabled for S/390). These are no longer marketed for EFS solutions.
- Various models of Multiprise 2000 systems. These are no longer actively marketed.

The following table offers a few initial comparisons:

| | S/390 CPUs | Rel CPU Perf | Typical S/390 Storage in MB | Internal Disk Perform | Typical Internal Disk Capacity | S/390 Chans | Significant Features |
|---|---|---|---|---|---|---|---|
| ThinkPad/EFS | < 1 | 1.4 | 256 | Good | 32 GB + | None | Very portable! |
| Netfinity/EFS (small) | 1 | 1.8 | 512+ | Very Good to Excellent | 160 GB | Limited | Multiple S/390 instances portable |
| xSeries EFS | 3 | 5 | 512-2048 | Excellent | 200 GB+ | Limited | Multiple S/390 instances significant non-S/390 workloads possible |
| P/390<br>R/390<br>Int. Serv. | 1 | 1 | 256 fixed | Limited | 18-36 GB | Limited | Portable (except IS) |
| MP3000-P30 | 1 | 8 | 1024 fixed | Very good | 72 GB | Excellent | LPARs<br>Crypto hardware<br>ICMF |

| | S/390 CPUs | Rel CPU Perf | Typical S/390 Storage in MB | Internal Disk Perform | Typical Internal Disk Capacity | S/390 Chans | Significant Features |
|---|---|---|---|---|---|---|---|
| StarterPak model A10 | 1 | 4+ | 1024 fixed | Good | 90 GB | 4 ESCON | LPARs<br>Crypto hardware<br>ICMF<br>OSA-2 |

This table is intended only as a starting point for positioning a ThinkPad/EFS system and needs a number of qualifications:

- The relative CPU performance numbers shown are *rough* indications of processor performance. Total system performance is dependent on many other factors as well, including memory size and disk performance. Also, processor performance depends on the nature of the workload and may vary *considerably* from the indicated ratios. Do not attempt to use this table for anything other than positioning the EFS models listed relative to the other specific machines shown.
- ThinkPad/EFS is shown as less than 1 CPU. This indicates that the single PC processor in the ThinkPad must be shared between Linux processes (such as S/390 emulated I/O) and S/390 instruction emulation.
- An x/Series 430 EFS system can have 1, 2, or 3 processors enabled for S/390 processing.
- Many of these machines can have a wide range of memory installed. The numbers shown are intended to represent typical systems.
- All the machines except the StarterPak can have a wide range of disk storage installed. Again, the numbers shown are intended to represent typical installations. The numbers represent effective disk capacity, after RAID or mirroring overhead.
- The S/390 Chans column refers to S/390 channel connectivity, using ESCON and parallel channels. All of the systems offer some channel connectivity, but all except the MP3000 are quite limited in this area.
- This table ignores RAS characteristics. The PC-based machines (Netfinity/EFS, ThinkPad/EFS and all the P/390-based machines) are designed to different RAS standards than the more *industrial strength* MP3000 and StarterPak systems. This may be an important factor when selecting a system for essential 24x7 production requirements.

## 1.3 ThinkPad/EFS hardware used

ThinkPad/EFS systems are provided through IBM business partners. They cannot be ordered directly from IBM. Different business partners may start with slightly different base configurations and offer different options. The system we describe in this redbook should closely match most starting configurations offered by business partners.

The basic hardware requirements are:

- A Thinkpad in the A21p, A22p, A23 (or T23) series, or one that is closely compatible, is the only supported base at the time of writing.
- A major compatibility requirement is that the X windows support in Red Hat Linux 7.1 must operate correctly with the ThinkPad display.
- It must have a USB port (for the FSI license *dongle*).
- It should have 512 MB memory (or more, if available).
- It should have at least one 32 GB (or larger) disk drive.
- The option for a second 32 GB (or larger) drive (swappable in the slot used by the CD-ROM drive and the diskette drive) should be considered.

- It should probably have the integrated Ethernet adapter, and possibly a second (PCMCIA) Ethernet adapter or a PCMCIA token ring adapter (depending on your requirements).

ThinkPad models change frequently and the above requirements may need to be adjusted to match current offerings. The author used a T20 ThinkPad (which met the other requirements) and had no problem with it. However, from a formal support viewpoint, only selected ThinkPad models should be used. At the time of writing, these were the A21p and A22p machines.

No particular disk drive size is required. A system needs about 1.5 GB for Linux. Most of the remaining disk space can be used for emulated S/390 drives. An emulated drive takes approximately the same number of bytes on the ThinkPad disk as are available on a “real” drive. A 3390-3, for example, requires about 2.8 GB on the ThinkPad disk. It is convenient to have extra space—say 2 GB—left for a work area.

Using two disk drives allows some overlap of drive operations. We did not attempt to measure this and we suspect that the effect is small. FLEX-ES provides very effective cache functions for emulated drives and this greatly reduces the actual disk drive I/O rates (especially seek rates) compared to “real drives” on “real S/390s.”

## 1.4 Terminology

EFS descriptions can become confusing if the terminology is not well defined. These are important terms:

- **Processor** means a PC processor in the ThinkPad (or a Server processor in larger platforms).
- **Server** means our underlying ThinkPad hardware.
- **Server operating system** means Linux for the ThinkPad/EFS system described in this redbook.
- **S/390 CPU** (or simply **CPU**) means an emulated single S/390 CPU engine.
- **S/390 system** means an emulated S/390 (in the EFS system) that might have more than one emulated S/390 CPU engine.
- **Instance** (or **FLEX-ES instance**) means an emulated S/390 system. We can run several FLEX-ES instances if we have sufficient resources.
- **OS/390** means the S/390 operating system. We could also use VM/ESA or VSE/ESA, but we elected to work only with OS/390 and z/OS (31-bit mode).
- **ThinkPad/EFS** is a generic name. The business partners providing this product often have their own names for their specific implementation of the product.

For any *FLEX-ES instance*, there is a one-to-one correspondence between *processors* enabled for S/390 and *S/390 CPUs*. Running multiple *instances* is something like running multiple OS/390s in separate LPARs (but is not quite the same). We can run multiple *instances* of *single-CPU S/390s* in a ThinkPad/EFS system.

2

# System and Linux installation

ThinkPad/EFS installation, with Linux, is much simpler than installation of other EFS platforms. There are three primary reasons for this:

- The appropriate ThinkPad models can be used “out of the box.” We are unaware of any requirements for BIOS upgrades.[1] There are no RAID functions to customize.
- The server operating system (Red Hat Linux 7.1) is installed “out of the box.” We are unaware of any required patches or fixes essential for EFS use.
- We can use simple Linux files for emulated S/390 disks.

All of these factors are subject to change, of course, and are likely to change with future systems. Nevertheless, we expect ThinkPad/EFS installation to remain simpler than installation of other EFS systems. A background consideration is that, because ThinkPad/EFS is not intended for production use, there is slightly less emphasis on using the latest BIOS fixes, latest Linux fixes, and most optimum disk performance techniques.

We elected to have Linux as the only PC operating system on our ThinkPad. You could create multiple primary partitions, install a boot manager of your choice, and install a Windows system in addition to Linux. This should be transparent to FLEX-ES, assuming you manage to install a functional Linux.

## 2.1 Disk planning

A little background on FLEX-ES disk techniques may be interesting, especially for those who have used or read about other EFS platforms. FLEX-ES uses *raw disks* on its other platforms. Using a UnixWare[2] base (as used with current Netfinity/EFS systems) for discussion, the following elements are involved:

- A raw disk is contiguous space on a hard disk. (This statement ignores the effects of RAID striping. The Server operating system sees a raw disk as contiguous space if a hardware RAID adapter is used.)

[1] This is not to say that you should not install BIOS upgrades. We are unaware of any that are *required* for using FLEX-ES.

[2] These comments also apply to Netfinity/EFS with an Open UNIX 8 base. Open UNIX 8 is the replacement release for UnixWare 7.1.1.

© Copyright IBM Corp. 2001

- A raw disk (or several raw disks) is required for each S/390 DASD volume being emulated. For example, if an OS/390 system requires 15 3390 volumes (for the system and user volumes), then FLEX-ES would need at least 15 raw disks in UnixWare.
- A raw disk has no UNIX file system. It can be used (by UNIX programs) as a single, sequential file. Multiple raw disks are tedious to administer and can lead to fragmentation issues if not well planned. This disk management has often been the most complex element of EFS installation on other platforms.
- Raw disks are used by FLEX-ES because UnixWare does not buffer access to them. Instead, FLEX-ES code does all the buffering using designs that are optimized for emulated S/390 volumes. This provides a major performance boost for FLEX-ES.[3]
- In principle, FLEX-ES could use standard UNIX files to emulate S/390 volumes. However, the normal UNIX buffering is not well suited to this emulation and the resulting performance is poor.[4]

Linux has added more factors to these elements:

- Linux has raw disks, but buffers I/O for them. This means that the unique FLEX-ES coding for raw disks is not very effective.
- Linux I/O handling for normal file systems is generally faster than that of traditional UNIX systems.
- Linux directly supports large files (larger than 2 GB), as required for 3390-3 and 3390-9 emulation.

For these reasons, FLEX-ES uses normal Linux files to emulate S/390 DASD. It is possible to use Linux raw disks, but the performance benefits are slight and not worth the administration efforts involved. Linux has recently begun to support *raw devices*. It is possible that future versions of FLEX-ES may use these if performance benefits warrant it. For this redbook, and for current ThinkPad/EFS systems, we will use only normal Linux files for emulating S/390 disk volumes. This *substantially* simplifies installation and administration of the EFS system.

We plan to use a simple naming convention with names such as /s390/OS39RA, for the Linux file containing an emulated 3390-3 with volser OS39RA.

## Differences

Typically, under Linux, an emulated S/390 DASD volume is a single Linux file. In this case (a single file), the file can have any convenient name. No FLEX-ES naming convention is required.

Under UnixWare (for a Netfinity/EFS system) emulated S/390 DASD volumes may occupy several raw disks.[5] In this case (multiple UNIX files per S/390 volume) a special FLEX-ES naming convention must be followed. This naming convention requires UNIX file names to end with a lowercase ASCII “s” followed by a numeric digit.

[3] This is not unique to FLEX-ES. Other major UNIX middleware packages, such as some relational data base managers, also use raw disks for the same reasons.
[4] This does not imply that UnixWare disk handling is poor. It is very good for normal UNIX applications. Emulation of S/390 volumes is a very specific, narrow application that does not match typical UNIX file usage.
[5] This is not required, can be done to reduce potential fragmentation issues with raw disk space.

## 2.2 ThinkPad

Before installing Linux, we installed all our ThinkPad options: additional memory, internal Ethernet port, the CD-ROM drive, and a 32 GB disk drive[6] in the standard disk position. We did not install our Ethernet PCMCIA card at this time. (In fact, we never used the PCMCIA adapter because we could successfully share the integrated Ethernet adapter between Linux and OS/390 TCP/IP.)

We started the system and selected F1 for BIOS setup. Working with BIOS functions, we verified that we had 512 MB of memory and that the internal Ethernet adapter was recognized. We set the time and date. In the *Startup* section, we verified that the *Boot device List F12 Option* was enabled; this lets us temporarily select the CD-ROM as a boot device. Our normal Boot List contained *Removable Devices* (that is, a diskette) followed by *Hard Drive*.

After exiting the BIOS setup function and performing power-off/power-on functions, we had an initial prompt to provide a temporary boot device list. Pressing F12 provided this list. The CD-ROM was the third element in the list and this is needed to start Linux installation.

## 2.3 Linux installation

We purchased a Red Hat Linux 7.1 standard package in a local store. (They also had a Deluxe version and a Server version; we selected the basic version.) This consists of four CD-ROMs. The first two contain the system; the third contains Linux source code and the fourth contains documentation.

We booted from the first Linux CD (using the ThinkPad F12 option to select a temporary boot device). This produced a Red Hat logo screen and offered the choice of *graphics mode* or *text mode* installation. We selected *text mode*, primarily because it was easier to document for this redbook. We then went through a number of installation prompts:

```
Language: English
Keyboard: US
Welcome to Red Hat Linux:  OK
System Type: Custom System
(Possible message about "Bad Partition Table"; if so, select Initialize)
Partition: Manually Partition
Select partition tool: Disk Druid
```

You can select **fdisk** instead of Disk Druid, and you should use whichever one you find most comfortable. We found that the Linux **fdisk** was just different enough from DOS **fdisk** to be confusing, so we used Disk Druid.

Our ThinkPad disk had no installed partitions and was 100% available for allocation. Using Disk Druid, we allocated four partitions:

```
Mount Point   Device  Requested  Actual   Type          Purpose
  /boot       /hda1     32 M       36 M  Linux native  boot partition
  /           /hda5   3000 M     3000 M  Linux native  Linux system
              /hda6    150 M      155 M  Linux swap
  /s390       /hda7      1 M    26123 M  Linux native  for S/390 volumes

OK (to exit from Druid)
Save changes: yes    (If this message appears)
```

[6] Business partners providing ThinkPad/EFS systems were switching to 48 GB drives at the time of writing. This change has no effect on the comments here, except that it makes a single-disk system more attractive.

We selected a completely arbitrary size, 150 MB, for a Linux swap partition. In the light of later experience, this could probably be smaller. However, it is a small fraction of our available space and appeared to be a safe choice. The Red Hat documentation indicated that 32 MB was the maximum possible boot partition size, so we selected that. Again, it could probably be smaller.

It may take some experimentation with Disk Druid to mark a partition as *Linux swap*. We suggest simply working with the Disk Druid options (Add, Delete, Edit) until you succeed.

The documentation did not provide much guidance for selecting the amount of space for Linux itself. We elected to put all of Linux into the root file system, and arbitrarily assigned 3000 MB for this.[7] (If you elect to install "everything" in RH7.1, you will need about 2.5 GB. Our more minimal installation actually used about 1.1 GB.) If you have more Linux or UNIX experience, you might want to create several file systems for Linux[8] instead of placing everything in a single root file system. Other than having about 25 MB available in /usr (for later FLEX-ES installation and working files), there are no special requirements for your Linux disk configuration.

We then assigned all the remaining space on this drive to a partition with the mount point /s390. This name is arbitrary, but clearly indicates the intended purpose of the partition.

At this point, we had only a single hard disk installed. Our second hard disk will eventually replace the CD-ROM drive; we cannot have both the CD-ROM drive and the second hard disk installed at the same time. We initialized the second drive later.

The installation prompts continued with fairly basic items:

```
Choose partition to format:  ALL  (place asterisk in all lines)
LILO Configuration, special parameters:  (leave blank)  OK
LILO Configuration, where to place boot record: MBR
LILO Configuration:  OK
     Device      Partition Type    Default    Boot
     /dev/hda5   Linux native        *        Linux
Network Configuration:
     [ ] use bootp/dhcp    (deselect this)
     IP address:           9.12.17.210
     Netmask:              255.255.255.0
     Default gateway:      9.12.17.150
     Primary name server:
     Secondary name server:
     Ternary name server:
Hostname Configuration:
     Hostname: tpefs1
Security level: No Firewall              (use the space bar to change it)
Mouse: 3 button PS/2 mouse
Language support: English
Time Zone: (as appropriate)
Root password: xxxxxxxx                  (must be at least 6 characters)
Add user: aaaaaaa                        (use your name)
User password: xxxxxxxx                  (must be at least 6 characters)
```

The *No Firewall* security option was appropriate for our purposes, but may not match your needs. As far as we know, this has no particular FLEX-ES security implications[9] and you can select the options appropriate for your needs. Select any user name you like for the *Add User* function, but do not select the name *flexes*. (*flexes* is a special name for FLEX-ES and will be

[7] We intentionally specified considerably more space than we needed. We assumed the extra space would be useful for a variety of purposes.

[8] Separate file systems for /tmp and /home are the most common alternatives.

automatically installed later.) The installation process will automatically create a group with the same name as the user name you specify, and add this user to the group. For example, if you specify username *ogden*, the installation process will automatically create group *ogden* and add user *ogden* to group *ogden*.

```
OK to Exit from the User Account routine
Authentication Configuration:
     [*] Use shadow passwords
     [ ] Enable MD5 Passwords
     [ ] Enable NIS
     [ ] Enable LDAP
     [ ] Enable Kerberos
OK
Package Group Selection: Customized     (you need to scroll the following list)
     total install size:  954 M
     [*] Printer support
     [*] X Window System
     [ ] GNOME
     [*] KDE
     [*] Mail/WWW/News Tools
     [*] DOS/Windows Connectivity
     [ ] Graphics Manipulation
     [*] Games
     [ ] Multimedia Support
     [*] Laptop support
     [*] Networked Workstation
     [ ] Dialup Workstation
     [ ] News Server
     [ ] NFS Server
     [ ] SMB (Samba) Server
     [ ] IPL/Netware (tm) Connection
     [ ] Anonymous FTP Server
     [ ] SQL Server
     [ ] Web Server
     [ ] DNS Name Server
     [*] Network Management Workstation
     [ ] Authoring/Publishing
     [ ] Emacs
     [*] Development
     [*] Kernel Development
     [*] Utilities
     [ ] Everything
OK
```

You can select gnome in addition to (or instead of) kde. These two packages are competing implementations of an X windows-based desktop environment. You need at least one of them. Experienced Linux users often have strong preferences for one or the other package. We had no strong preferences and arbitrarily selected kde; we have been quite satisfied with this choice. (We also used gnome on other installations, with no problems.)

[9] FLEX-ES needs to use TCP/IP port 24 on Linux for its Terminal Solicitor function. Almost all FLEX-ES installations will require this. FLEX-ES will also use port 555 if you have remote FLEX-ES resources installed. The typical ThinkPad/EFS user will probably not use this, especially for initial FLEX-ES use. If you install firewall functions, be certain that port 24 is not blocked.

You also need X windows, Laptop support, Networked Workstation, Network Management Workstation, Development, Kernel Development, and Utilities. You may want to change some of the options we selected or did not select. In particular, you might want to select *Anonymous FTP Server* (if you want *any* ftp server functions) and *SMB (Samba) Server* (if you want Windows connections). FLEX-ES does not require these, but they may be useful in many cases.

```
Some of the packages you...selected...require packages
you have not selected. ...
    (*) Install packages to satisfy dependencies
OK

X probe results: Video Card S3 Savage (generic)
                 X Server XFree89
OK (to accept this hardware)
Install log will be in /tmp/install.log
Now format file system(s)                (took about 1 minute)
Transferring install image to hard disk
Package installation:  425 packages to install
```

Installation took about 16 minutes. It called for the second CD-ROM when it was approximately half finished.

```
Create boot diskette: No                 (could not switch to diskette drive)
Monitor Setup:  IBM 9513 T55A TFT
                4 MB
                No clock chip
                24 bit 1024x768
     Starting X (test): OK
                Select "Yes" with mouse
                Select "Start X automatically"

Reboot                                   (ejects the CD)
```

Unfortunately, this Linux distribution does not have specific display parameters for ThinkPad displays. As a guess, we specified our ThinkPad display as an IBM T55A (which is a flat-panel LCD display) and this has worked well. As part of the monitor setup dialog (although it is more concerned with video adapter parameters) we specified 4 MB video memory, no clock chip (as recommended by the installation dialog) and 24 bit x 1024 x 768 resolution. While this may not be an optimum definition for ThinkPads, it works well and provides an excellent display.

After rebooting and logging into Linux as root, we tried the following:

```
# df -h                                  (display file system usage)
# cd /proc
# cat cpuinfo
# cat iomen
# cd /
# ls -al                                 (verify that normal root entries are present)
```

### 2.3.1 Installation notes

The *Custom System* installation option was required in order to have both X windows and xinetd active. The *Laptop Installation* and *Workstation Installation* choices both omit xinetd support, and the *Server Installation* choice omits X windows. We require xinetd functions and X windows functions and are forced to use the *Custom* installation option.

After the installation outlined above, you should be able to log into the system and **ping** another host on your subnet.[10] If the gateway IP system you specified is functional, you may be able to access remote hosts. You will note, however, that other systems cannot **telnet** to your ThinkPad Linux.[11] If you want to permit **telnet** or **ftp** into your machine, you need to do the following:

```
# /sbin/chkconfig telnet on                    (to permit incoming telnet)
# /sbin/chkconfig wu-ftpd on                   (to run an ftp server)
```

If this does not work for some reason, you can edit the relevant files as follows:

```
# cd /etc/xinetd.d
# vi telnet                            (you can use another editor, if you like)
  change "Disable = Yes"
  to "Disable = No"
  (save and exit from your editor)
Stop and restart your Linux system, OR force xinetd to restart:
# ps -ef | grep xinetd
  (find the line for xinetd and read the PID number)
# kill -s USR1 pidnumber               (the PID number for xinetd)
```

Similar editing of the wu-ftpd file in the same directory will enable the ftp server.

[10] You may have problems if you **ping** systems on other subnets. The remote system must have a defined route back to you in order for the **ping** to work. Testing with another system on your local subnet avoids routing problems.
[11] This is not required for FLEX-ES operation.

3

# FLEX-ES and OS/390 installation

This chapter describes the installation process for FLEX-ES. System definitions, resource definitions, and operation is described in the next chapter. This chapter also describes the installation of an OS/390 AD CD-ROM system. If you do not use these AD systems, you should skip this section and work with your IBM business partner to understand alternate methods for installing S/390 software.

© Copyright IBM Corp. 2001

# 3.1 FLEX-ES

This redbook concentrates on the use of FLEX-ES with a ThinkPad running Linux. Furthermore, we describe a particular ThinkPad/EFS configuration used as a starting point for IBM's S/390 Partners in Development (PID) organization. This configuration might be regarded as the minimum practical system for using OS/390 and z/OS.

The following descriptions and figures assume this particular ThinkPad/EFS configuration. Other configurations and more complex systems are possible, but are not described here.

## 3.1.1 Brief introduction

Conceptually, FLEX-ES can be viewed as the following components:

- A S/390 instruction emulator, which might be seen as the heart of the system. It examines each S/390 operation code and emulates that operation, using the instructions of its underlying PC processor.
- A *resource manager* that controls the interfaces between the emulated S/390 processors and emulated I/O devices and connections.
- Emulators for various S/390 I/O devices.
- A *FLEX-ES console* for controlling FLEX-ES startup and operation. (This is not related to the OS/390 master console.)
- A *Terminal Solicitor* program that emulates local, channel-attached, non-SNA 3270 terminals. The actual terminals are TN3270 sessions that connect to this program through normal TCP/IP protocols.
- A number of utility programs to help set up and run the FLEX-ES environment.

Figure 3-1 provides a simplified view of FLEX-ES operation. FLEX-ES is simply a process[1] under Linux. While this illustration should not be taken too literally, it can be used to make a number of basic points about FLEX-ES.

[1] It is actually a number of interrelated processes and threads, but that can be ignored at the conceptual level.

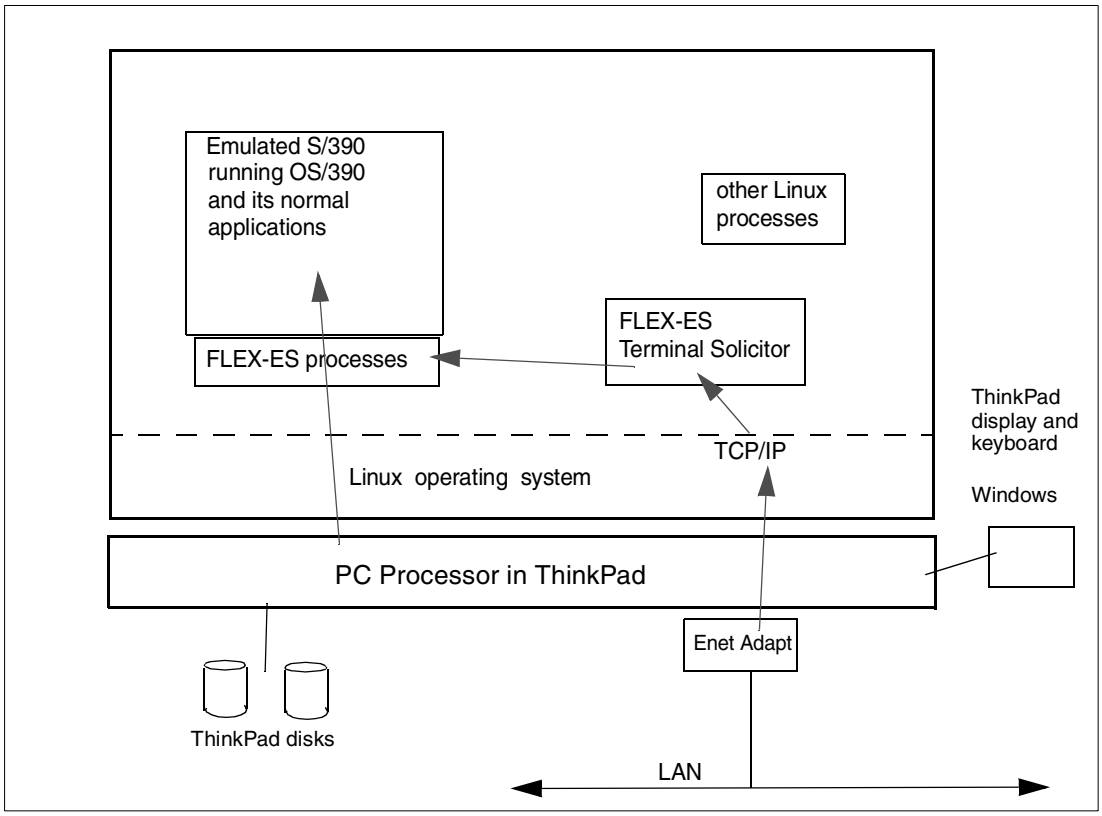


*Figure 3-1 Conceptual view of FLEX-ES system*

Key points include:

- FLEX-ES is a software product.[2] All the hardware shown (in Figure 3-1) is standard ThinkPad hardware.
- The FLEX-ES program, running under Linux on the ThinkPad's PC processor, emulates a complete S/390 environment.
  - FLEX-ES obtains sufficient virtual memory to emulate the "real memory" for the defined S/390 machine being emulated. While not detailed in the illustration, FLEX-ES can also emulate expanded memory.
  - FLEX-ES, as part of emulating a S/390, handles S/390 I/O instructions and emulates the S/390 I/O devices as required. In the illustration, various 3390 volumes (containing OS/390 and so forth) are contained on the ThinkPad disks.
- The FLEX-ES license specifies how many PC processors may be used (at any one instant) for S/390 emulation. For a ThinkPad/EFS system this will normally be one processor.
- The owner can elect to *dedicate* PC processor(s) to FLEX-ES S/390 use. This provides a performance boost. You cannot dedicate all the PC processors to S/390, because other Linux functions need access to a processor. If processors are not dedicated, then the normal Linux dispatching function will dictate which processors are being used by FLEX-ES at any instant. The number will never exceed the number permitted in the FLEX-ES license. A ThinkPad/EFS system has only a single PC processor, and it cannot be *dedicated* for S/390 emulation.

[2] The optional S/390 channel adapters available for use with FLEX-ES are ignored in this discussion.

- You can emulate a S/390 with a number of CPUs. This number cannot exceed the number of PC processors licensed for S/390 emulation. With the ThinkPad/EFS system we are describing, this means that only a single-processor S/390 can be emulated.
- The emulated S/390 can use LAN adapter(s) on the ThinkPad. OS/390 TCP/IP can run this way. *A single ThinkPad LAN adapter can be used both by the OS/390 TCP/IP and by Linux TCP/IP.* This differs from FLEX-ES under UnixWare, where a LAN adapter may not be shared by a S/390 TCP/IP and UnixWare.[3]
- The FLEX-ES Terminal Solicitor is a Linux process. It provides a TN3270 *server* function and listens (by default) on port 24 of Linux's TCP/IP. An external user can connect a TN3270 client to the Terminal Solicitor. The Terminal Solicitor and the FLEX-ES emulation process will then transform the TN3270 protocol so that it appears to be a local, non-SNA 3270 to the emulated S/390. This is the normal way to connect MVS consoles, TSO terminals, CICS terminals, and so forth. There is no particular limit to the number of terminals that can be connected this way.[4] Each one must have appropriate VTAM definitions for a local 3270, of course.
- If a LAN adapter is connected to OS/390 TCP/IP, then TSO terminals may connect this way. If OS/390 TCP/IP and UNIX System Services are configured for it, ASCII telnet sessions can also use this connection. The Terminal Solicitor is not involved in these connections. The two LAN interfaces would have separate IP addresses, of course, even though they may share a common LAN adapter.
- The illustration shows a single emulated S/390. It is possible to emulate several S/390s at the same time.[5] The PC processor(s) licensed for S/390 emulation will be dispatched among the several emulated S/390s. Only one Terminal Solicitor would be used; it can be shared by all emulated S/390s. The multiple emulated S/390s are known as multiple *instances* of FLEX-ES S/390 emulation. Multiple S/390 instances of OS/390 are probably not very practical with a ThinkPad/EFS system, due to limited ThinkPad memory and due to there being a single ThinkPad processor that must be shared among all the S/390 instances plus Linux. However, multiple VSE/ESA or VM/ESA systems might be practical on a ThinkPad/EFS machine.
- While it is not apparent in the illustration, the amount of real PC memory available is a key performance factor. The performance of an emulated S/390 degrades quickly if Linux performs much paging (especially if the paging is triggered by FLEX-ES processes or if the memory being paged is used by FLEX-ES to emulate S/390 real memory). The recommended paging level for Linux is zero once steady-state S/390 emulation is running, and should seldom rise to more than one page per second. OS/390 paging (in the emulated S/390) might be much higher than this, with no ill effects. Do not confuse Linux paging with OS/390 paging. The amount of real ThinkPad memory available is a key factor in setting the size of the emulated S/390 machine memory and in deciding whether to use multiple S/390 instances.
- The terminal shown in the figure is usually the ThinkPad display and keyboard. The typical ThinkPad/EFS usage would be through X Windows. In this mode, several windows can be open. Some of these might be x3270 sessions (TN3270 clients) connected to OS/390 through the Terminal Solicitor; one of these sessions might be the MVS master console. Another window might have FLEX-ES command line interface (CLI) prompts used to control FLEX-ES operation. CLI commands are used to IPL a S/390 operating system, for example.

---

[3] It also differs from P/390-based machines and the MP3000, none of which permit sharing of LAN adapters by multiple TCP/IP stacks.

[4] There is an overall FLEX-ES limitation of 2048 emulated devices, but a ThinkPad/EFS system is unlikely to approach this limit.

[5] A different FLEX-ES license is not required for this. The FLEX-ES license indicates how many Server processors may be used for S/390 operation, but does not limit how many instances of S/390s you can run.

### 3.1.2 Installation

FLEX-ES can be delivered a number of ways:

- FTP over the Internet
- CD-ROM
- Diskettes
- Preinstalled by a business partner

The FLEX-ES package for ThinkPads is not large (about 3.5 MB), and an FTP download is easy, even over a typical dial-up line. In addition to the FLEX-ES code, you need a FLEX-ES license. This is a few hundred bytes and will normally be shipped with your system. We expect most business partners will burn a CD with the FLEX-ES packages and we will use this for our illustrations.

FLEX-ES for ThinkPad/EFS is shipped as three rpm packages, plus a license key file. (You need a USB dongle that matches the license key file, of course.) Our basic installation consisted of the following commands:

```
# mount /dev/cdrom /mnt/cdrom                    (CD with FLEX-ES and key file)
# cd /mnt/cdrom
# ls -al
   (you should see the three rpm files and whatever other files your business partner
   included.  We placed our key file on our CD)
# rpm -i flexes-6.0-5.i386.rpm
spawn passwd flexes
Changing password for user flexes
New UNIX password:
BAD PASSWORD:  it is too simplistic/systematic
Retype new UNIX password:
passwd: all authentication tokens updated successfully
# rpm -i msgmgr-6.0-5.i386.rpm
# rpm -i ftlib-6.0-5.i386.rpm
# cp /mnt/cdrom/yourlicensekeyname.key  /var/adm/flexes/.flexeslicense
```

Ignore the error messages about passwords. The installation process creates userid *flexes* with the password *abcdef1*. It also creates a group named *flexes* and makes user *flexes* a member of this group. You can later change the password to anything you like.

We used a test release of FLEX-ES for ThinkPad/EFS, as reflected in the version numbers that are part of the rpm package names. Your rpm names will be different but should have the same pattern. Likewise, the file name for your license key will be different, but should be easily recognizable.

In practice, most FLEX-ES management and operation is from userids *root* and *flexes*. Of course, the standard advice applies that you should log into the system as *flexes* (or some other id) and then **su** to *root*. This **su** step complicates documentation, so we will simply discuss *root* and *flexes* usage and assume you become *root* in a proper way. In our documentation a # prompt indicates *root*, and a $ prompt indicates a non-root userid.

FLEX-ES installation creates */usr/flexes* and makes this the home directory for userid *flexes*.

We then shut down Linux, connected the USB dongle, rebooted and did this:

```
# cd /usr/flexes
# mkdir rundir                                (conventional location for FLEX-ES files)
# vi /root/.bash_profile
   USERNAME="root"
   PATH=$PATH:/usr/flexes/bin                 <=== add this line
   export USERNAME BASH_ENV PATH
```

```
# vi /usr/flexes/.bash_profile
   USERNAME="flexes"
   PATH=$PATH:/usr/flexes/bin              <=== add this line
   export USERNAME BASH_ENV PATH
```

Adding /usr/flexes/bin (where all the FLEX-ES executables are located) to PATH makes operation easier.

### Installing the FLEX-ES license key

You need a FLEX-ES license key, which is a Linux file of a few hundred bytes. Your IBM business partner should obtain this for you. The business partner will need to obtain a USB *dongle* along with the license key. The dongle is a small hardware device that plugs into the ThinkPad USB port and contains an encrypted serial number that is reflected in the license key.

Copy the license key file to */var/adm/flexes/.flexeslicense*, as noted in the instructions above. (Note that the target file name begins with a period.) If the license key is provided on diskette you will need to copy it from there. For example:

```
(shut down Linux)
(remove the CD-ROM drive from the Ultrabay of the ThinkPad)
(insert the diskette drive in the Ultrabay)
(boot Linux and log in as root)
# mdir a:
(if the diskette is in DOS format, you should see the file with your key)
# mcopy a:your.key  /var/adm/flexes/.flexeslicense
(done)
(if the diskette is not in DOS format, try the following)
# mount /dev/fd0 /mnt/floppy
# ls -al /mnt/floppy
(if the key file is displayed, copy it with the following command)
# cp /mnt/floppy/yourlicensekeyname.key  /var/adm/flexes/.flexeslicense
```

ThinkPad models are updated and changed frequently. The exact method of changing from CD-ROM use to diskette use depends on your ThinkPad model. In some models, you can have both a CD-ROM drive and a diskette drive active at the same time. In this case, there is no need to shutdown the Linux to swap drives.

If you have multiple FLEX-ES licenses, be certain to identify the correct license with the correct dongle.

## 3.1.3 X3270

A version of **x3270** (a TN3270 client) is distributed with FLEX-ES. A different version of **x3270** is distributed with Red Hat Linux 7.1. You should use the version distributed with FLEX-ES, even though it may be older than the version distributed with Linux. Since we placed the FLEX-ES libraries *after* the Linux libraries in our PATH environment, the system will find the Linux **x3270** first. You can correct this in a number of ways. We did this:

```
# cd /usr/bin/X11
# mv x3270 x3270xxx                          (rename it to something obscure)
```

After doing this, the FLEX-ES distribution of **x3270** will be found using our PATH. (You could accomplish the same thing by placing /usr/flexes/bin first in the PATH, or you could simply delete the X11 version of **x3270**.)

### 3.1.4 Next steps

We created mount point /s390 when we installed Linux. It is the large file system that occupies most of our hard disk. We intend to use a second hard disk (in the Ultrabay slot of the ThinkPad) and we can create a mount point for it now. (We will initialize and mount the second hard disk later.) Both mount points should be owned by userid *flexes* because they will be used to hold emulated S/390 volumes:

```
# mkdir /s391                              (mount point for second hard disk)
# chown flexes:flexes /s390
# chown flexes:flexes /s391
```

The names *s390* and *s391* are arbitrary, of course, but they express the purpose of the file systems.

You next need to install your S/390 operating system. We used an AD system; this is described next. If you use another installation method, you need to work with it now. After the S/390 operating system is installed, you need to create matching FLEX-ES definitions and then test the system. This is described in Chapter 4, “FLEX-ES Operation” on page 27.

## 3.2 OS/390 AD systems

In principle, you can install OS/390 on your ThinkPad/EFS using any of the distribution media and logical packaging available from IBM.[6] In practice, most of IBM’s S/390 Partners in Development (PID) members use a CD-ROM package that is available only to members of their group who obtained a system through the PID organization. At the time of writing, IBM does not provide a more general OS/390 or z/OS distribution packaged on CD-ROM.

### 3.2.1 AD systems

We chose an OS/390 AD CD-ROM system (OS/390 release 2.10) for the ITSO projects used to produce this redbook. Not all readers are familiar with these “AD” systems; we briefly explain them here.

An AD (Application Development) system is a prepackaged OS/390 (or z/OS), with a number of priced features and additional program products included.[7] Considerable customization has already been done, making the system immediately usable for many functions. The AD systems are available only to members of IBM’s PartnerWorld for Developers (formerly known as Partners in Development, or PID) who obtain systems through the PID program. They are not available to general IBM customers.

Why did we use it for our EFS projects? We used it primarily because it provides a very easy way to install a useful OS/390 system. We could have built an OS/390 system starting with a ServerPac, in the same way most OS/390 customers build their systems. However, this requires considerably more time and effort and would have detracted from the time spent working with EFS elements. It would also require a S/390 with tape drives compatible with the ServerPac media, and the ThinkPad does not meet this requirement.

In general, an OS/390 AD system is a rather straightforward implementation of OS/390 and contains no magic components or “clever” setups. The experience and results of using it on ThinkPad/EFS should be about the same as using any other straightforward OS/390 implementation.

---

[6] This statement assumes you have a tape drive on your ThinkPad/EFS that is compatible with the IBM-provided media, of course.

[7] There are AD systems available for VM/ESA and VSE/ESA also. The discussion in this chapter is about OS/390, so we limit this discussion to the OS/390 AD systems.

The AD CD-ROM systems, as the name implies, are distributed on CD-ROMs. This aspect is not common to other OS/390 packaging, but does not affect the characteristics of the system once it is installed. The CD-ROMs are not seen by the S/390; they are processed[8] by a server (usually with OS/2, but with Linux in the case of a ThinkPad/EFS system) that routinely handles CD-ROM drives.

## 3.2.2 OS/390 on CD-ROM

The CD-ROM versions are based on the same versions available (on tape) through the standard IBM software distribution processes. There are no modifications involved due to the use of CD-ROMs for distribution. However, the PID versions distributed on CD-ROM are considerably more customized than the standard IBM software distribution versions.

In general, the PID versions (once installed on the S/390 system disks) are immediately ready for use. A set of userids is provided, for example. Minor additional customization (such as setting IP addresses) is required, but very little work is required compared to, for example, a ServerPac distribution of OS/390. The penalty for this immediate usability is that many configuration and customization decisions have been made by IBM. In general, the resulting systems are quite suitable for a small development organization, but would probably not be suitable for a large, highly-structured production installation. Since the target for the PID CD-ROM systems is smaller development organizations, the PID CD-ROM systems have been very well received.

### Basic CD-ROM formats

The fundamental format of the CD-ROMs is PC-compatible. That is, any DOS, OS/2, or Windows operating systems (and, as far as we know, most PC UNIX and Linux operating systems) can recognize the files and directories on the CD-ROMs. The CD-ROMs typically contain README files (in ASCII), P/390 DEVMAP files (binary), an OS/2 UNZIP program (binary), along with an AIX UNZIP version (binary), and one or more files containing the OS/390 volumes (binary). From a PC viewpoint, these OS/390 volumes are binary files and are usually very large files.

The OS/390 volumes (on the CD-ROM) are ZIPed files in AWSCKD format. AWSCKD is a P/390 device manager program that emulates 3380 and 3390 devices. In general, a complete 3390 volume is in one or two PC files. All the PID releases of OS/390 are in this format.

The AWSCKD format is, in essence, a complete image of a S/390 disk volume. Within an AWSCKD file CKD tracks, cylinders, R0s, and so forth are defined. There will be a standard label, a VTOC, probably a VTOC index, and whatever data sets appear on that volume in an OS/390 context. The data is in S/390 format. Text contained on a S/390 volume is EBCDIC and executables are S/390 binary files suitable for execution by OS/390. To a PC program, an AWSCKD file is simply a large binary file that is not useful to typical PC programs.

AWSCKD files on the CD-ROM are in ZIP format simply to save space. It is usually possible to ZIP an AWSCKD 3390-3 (2.8 GB) so that it fits on a CD-ROM (about 600 MB). There is no basic requirement that ZIP files be used and, in some cases, disk images might not be zipped.

[8] This is true when installing on a P/390, Integrated Server, or Multiprise 3000. The processing consists of UNZIPing PC files. Each file contains an emulated 3390 disk volume.

The AWSCKD format was developed for P/390s where the underlying operating system used to emulate CKD drives is OS/2. OS/2 is a 32-bit operating system and has the usual restriction that a single file cannot be larger than 2 GB. An AWSCKD-emulated 3390-3 requires more than 2 GB, and is split into two AWSCKD files, the first is 2 GB and the second is about .8 GB[9]. The two files are usually placed in a single ZIP file.

### FLEX-ES formats

FLEX-ES can handle the CD-ROM format discussed above. FLEX-ES emulates CKD drives, but the internal format (in the Linux files used for emulation) is different than the AWSCKD format. FLEX-ES provides a utility (**ckdconvaws**) to convert AWSCKD format to the FLEX-ES format for emulated CKD drives.

## 3.2.3 OS/390 device configuration

For the OS/390 R10 AD system, the first four volumes (first four CD-ROMs) contain an IPLable system on 3390-3 volumes, and we installed only these volumes. We also created three 3390-1 volumes for work space.

The AD system is already customized in many ways, including IODF device definitions. It is not the purpose of this redbook to discuss general AD system design.[10] The following sections mention specific S/390 device numbers ("addresses") such as A80 for the IPL volume. These addresses are included in the AD system; that is, they are included in the IODF distributed with the system.

We decided our initial OS/390 system (using the AD CD-ROM) would use the devices listed here. This represents a basic but IPLable OS/390 AD CD-ROM system.

| Address | Device | VOLSER | FLEX-ES Name | Comments |
|---|---|---|---|---|
| A80 | 3390-3 | OS39RA | /s390/OS39RA | IPL volume; OS/390 libraries |
| A81 | 3390-3 | OS3RAA | /s390/OS3RAA | More OS/390 basic libraries |
| A82 | 3390-3 | OS39M1 | /s390/OS39M1 | Paging, parmlib, catalogs, etc |
| A87 | 3390-3 | OS39HA | /s390/OS39HA | System HFS data sets |
| 560 | 3480 | | | FakeTape emulating 3480 |
| 700 | 3270 | | mstcon | NIP & OS/390 master console |
| 701-702 | 3270 | | L701 - L702 | Local non-SNA VTAM terminals |
| E20, E21 | CTC | | eth0 | Used for LAN TCP/IP interface |

The AD system can use a much larger set of addresses and devices than shown here.[11] These represent a basic useful system. Some of the addresses are arbitrarily chosen from the larger set provided with this AD system. The AD volumes containing the DLIBs, DB2, CICS, and IMS are not included in this list; they are not necessary for basic operation and we decided to not install them for our initial ThinkPad/EFS work.

---

[9] The two files contain the characters _1 and _2 as the last two characters of each file name. P/390 utilities recognize that the second file is a continuation of the first. Unlike FLEX-ES, the P/390 does not emulate 3390-9 drives. If it did, it would use four 2 GB files plus a 1 GB file to equal the 9 GB contained on a 3390-9 drive.

[10] A fairly detailed description of recent OS/390 AD systems is given in *S/390 Partners in Development: OS/390 (and z/OS) New User's Cookbook*, SG24-6204.

[11] This means that the IODF distributed with the AD system contains a large number of defined devices and addresses. We selected a subset of these already-defined devices and addresses.

The Linux file names shown in the table are not part of the AD system. Rather, they are the names we decided to use when installing these devices on our ThinkPad/EFS system. They are shown here for completeness. A important factor in our setup is to have an IPLable system on the primary ThinkPad hard disk. Although we will later install a second hard disk, we want to be able to IPL OS/390 without the second disk installed.

At the system level, we defined a S/390 system with 256 MB central storage and 64 MB expanded storage. The expanded storage size was completely arbitrary. We wanted our total system definition to be small enough to avoid Linux paging. We could have defined larger central storage, or no expanded storage, or a different combination of storage sizes.

## 3.2.4 An unzip program

A PC-compatible *unzip* program is included in the Red Hat 7.1 distribution. The command-line options are slightly different than those used with DOS-oriented unzip programs.

## 3.2.5 Installation tasks

We needed to perform the following tasks to install our OS/390 system:

1. Decide where to place the files containing the emulated volumes.
2. Unzip the required CD-ROM files and convert them to the FLEX-ES CKD format while loading them on disk.

These steps are described in some detail in the following sections.

### Unzipping and installing AWSCKD CD-ROM files

The installation of AD CD-ROM systems has been greatly simplified in the most recent FLEX-ES release.

We found that the easiest way to do the following process is to open two command-line windows on the Linux desktop:

- One window for the **unzip** and **ckdconvaws** commands, as userid *flexes*
- One window for **mount** commands and other utility commands, with an **su** to *root*

Having separate windows for the two major commands is convenient because *root* authority is needed for **mount** while the other functions should be under userid *flexes*.

#### *Mount CD*

Mount the first AD CD-ROM:

```
# mount /dev/cdrom /mnt/cdrom                    (issue from root)
$ ls /mnt/cdrom/os390
devmap.mvs   ickdsf.ipl   os39ra.zip   sadss.ipl
devmap.nme   migrate.doc  readme.mvs
```

#### *Unzip and convert a volume*

The output of the **unzip** command can be piped directly to the ckdconvaws command:

```
$ unzip -p /mnt/cdrom/os390/os39ra.zip | ckdconvaws -r - /s390/OS39RA 3390-3
FSIDC146 Max head=14, cyl=3343, blks=57
FSIDC180 cylinder nnn Completed in mmm milliseconds
FSIDC190 CKD conversion completed (3339 cyls copied, 0 cyl ignored)
```

The **ckdconvaws** command converts the AWSCKD format to the FLEX-ES format, and writes to the Linux file indicated.[12] The volume being unzipped is a 3390-3 (and we must tell the **ckdconvaws** program this). The second operand, /s390/OS39RA, is the target location for the Linux file that will emulate the S/390 volume. Your disk layout may be different, and you need to adjust this path and name as required for your system.

The nnn and mmm numbers shown will increment as the program runs. When the volume is finished, use the *root* window to issue the command:

```
$ cd /                                  (in the flexes window)
# umount /mnt/cdrom                     (in the root window)
```

3390-3 volumes are held in two files in the original AWSCKD format. Both these files are usually placed in the same *zip* file. The **ckdconvaws** program handles this transition automatically.

We can pipe directly from the **unzip** program to the **ckdconvaws** program only if the emulated volume will be held in a *single* file, as we are doing here. If the volume will be held in several files, a *named pipe* must be used. This is more common with UnixWare or Open UNIX systems and is described in the Netfinity/EFS redbook.

### *Load the other CDs*

Repeat the same process (**mount**, unzip/convert, **umount**) for the other CDs that are to be loaded. For the OS/390 R10 system, we used the following:

```
CD     CD File.................       Linux File      Disk Model
2     /mnt/cdrom/os390/os3raa.zip    /s390/OS3RAA      3390-3
3     /mnt/cdrom/os390/os39M1.zip    /s390/OS39M1      3390-3
4     /mnt/cdrom/os390/os39ha.zip    /s390/OS39HA      3390-3
```

## 3.3 File ownership

All of the files containing emulated S/390 volumes *must* be owned by userid *flexes* and groupid *flexes*. You can verify this, as follows:

```
# ls -al /s390                          (check the owner and group names)
# ls -al /s391                          (if you have a second disk)
(if you need to change ownership, do the following)
# su                                    (change to root, if not already root)
# chown flexes:flexes /s390/OS39RA      (or whatever file needs to be changed)
```

The “flexes:flexes” parameter format causes both the owner and the group name to be changed.

## 3.4 Minor problems

While working with various AD releases, we found two unusual situations.

[12] The -r option indicates that **ckdconvaws** should create the output file, if it does not already exist.

### Files unzip in wrong order

Unzipping one of the 3390-3 volumes (OS3RA9) for the OS/390 R9 AD system produces the two files in reverse order. (The R10 system did not have this problem.) In OS/2 terms, it unzips OS3RA9_2.A81 and then OS3RA9_1.A81. For an OS/2-based system this does not matter, since these are really two separate files (that together constitute a single 3390-3 volume). For older FLEX-ES releases, these files must be fed to **ckdconvaws** *in the correct order*, and this involves using intermediate files.

In the most recent FLEX-ES releases (6.1 or later), **ckdconvaws** handles the reverse-order files automatically.

### Volume in two separate zip files

Many of the AD volumes are 3390-3 volumes. An emulated 3390-3, on the OS/2-based emulation systems used to build the AD releases, requires two files. (This is due to the 2 GB maximum file size on OS/2.) The two files are normally placed in the same zip file on a CD-ROM. The **ckdconvaws** program, when used as documented in this redbook, combines the two files (contained in the single zip file) and produces the single emulated 3390-3 file used by FLEX-ES.

In rare cases, the zip file containing the two OS/2 files is too large to fit on a CD-ROM. (This happened with the OS39DA volume for the OS/390 R10 AD system.) When this happens, the two OS/2 files are zipped separately and placed on different CDs. (For R10, the larger file was placed on CD #5, by itself, and the smaller file was placed on CD #3, which also held the OS39M1 volume.) For OS/2-based systems (such as P/390s and emulated I/O on the MP3000), this is not a problem because the two files are unzipped and restored as two separate files.

In the latest FLEX-ES releases (6.1 or later), the **ckdconvaws** command will handle the two zip files separately and produce the correct results. That is, you would run **ckdconvaws** twice (once for each zip file) and point to the same output file name for your S/390 emulated volume. This is a considerable improvement over earlier releases, where a moderately complex procedure was needed to handle this situation.

## 3.4.1 Next steps

After loading the operating system, we need to make various FLEX-ES definitions and then test the system. This is described in Chapter 4, “FLEX-ES Operation” on page 27.

4

# FLEX-ES Operation

Installing and running a ThinkPad/EFS system requires a number of steps, some of which have already been described:

- Install the system hardware (described in “ThinkPad” on page 9).
- Install Linux as the server operating system (described in “Linux installation” on page 9).
- Decide on your disk configuration (described in “Disk planning” on page 7).
- Install FLEX-ES (described in “FLEX-ES” on page 16).
- Load OS/390 (many variations are possible, and you might use another S/390 operating system; our steps are described in “OS/390 AD systems” on page 21).

We have completed the steps listed above. The following steps are required to run a S/390 instance:

- Create FLEX-ES system and resource definition files.
- Compile the definition file, creating *rescf* and *syscf* files.
- Start **`resadm`**, the resource manager, pointing to a *rescf* file.
- Start **`flexes`**, the S/390 emulator, pointing to a *syscf* file.
- Start the **`flexescli`** program, which provides a command line interface for controlling a FLEX-ES instance.
- Start at least one TN3270 session (for the MVS master console).
- Issue an **`ipl`** command through the flexescli interface.
- Operate OS/390 as usual.
- Shut down the system when finished.

These steps are described in this chapter. In general, most of the setup described here should be done under the *flexes* userid. By convention, all the FLEX-ES configuration files, resource definitions, shell scripts, and so forth, are kept in /usr/flexes/rundir. This is not required, but it is common practice and the following examples use this directory.

© Copyright IBM Corp. 2001

## 4.1 FLEX-ES system and resource definitions

Before the newly installed/restored OS/390 system can be used, we must define the hardware and system resources to the FLEX-ES Resource Administrator. Appendix A, "FLEX-ES definition listings" on page 69 shows the input file that we used to define both the *system* and *resources* for our OS/390 system. This definition file was placed in /usr/flexes/rundir/defA1. (File name *defA1* is completely arbitrary.) We specify the name of this file as an argument for the FLEX-ES configuration compiler:

```
$ cd /usr/flexes/rundir
$ cfcomp defA1
Start FLEX-ES Configuration Utility
Configuration processing *SUCCEEDED* with no errors
Data Space Manager Terminated
```

We could issue the **cfcomp** command directly because we earlier placed /usr/flexes/bin in our Linux PATH. If you did not do this, you would need to use a full path name with the command:

```
/usr/flexes/bin/cfcomp defA1
```

Our FLEX-ES definition file consists of two sections: the *system* section and the *resources* section. The compilation creates files S10A.syscf and R10A.rescf, based on the names contained in our definitions.[1] We can then invoke the resource administrator to activate our resources:

```
$ su
Password:                   (<--- enter root password when this prompt is shown)
# cd /usr/flexes/rundir     (if you are not already there)
# resadm -s R10A.rescf      (<--- To activate our resources. Must be root)
# resadm -r                 (<--- To check that resources are started)
# exit                      (leave root)
```

Again, we could issue the **resadm** command directly because /usr/flexes/bin is in our PATH. You must be *root* to issue these **resadm** command options.

### Comments

The short steps just shown, to compile system and resource files and to activate the resources (but not the system), require some explanations.

Our definition file source had been placed in /usr/flexes/rundir. We created the *rundir* directory just after we installed FLEX-ES; it was not created automatically. Two separate definitions are needed for a FLEX-ES emulated S/390. One defines the *system* being emulated:

- Central memory size available to this system
- Expanded memory size
- Number and type of CPUs
- Number and usage of channels
- Control units for all system devices

The other defines *resources* for a single system or multiple systems, such as:

[1] We could have used two definition files, one for the system definitions and one for the resource definitions. The use of separate system and resource definition files is most common when multiple instances of FLEX-ES are used (meaning multiple S/390s being emulated at the same time). This is unlikely to be the case with a ThinkPad/EFS system. If we did use separate system and resource definition files, we would need to compile both of them in order to obtain the *syscf* and *rescf* files we need to actually use a S/390 emulation instance. We found it more convenient to always combine the definitions in one source file.

- Interfaces for all the control units defined in the system section
- Devices for all the control units defined in the system section

These two definitions can be two separate files or combined into one file. We elected to combine them into one file, as listed in Appendix A, “FLEX-ES definition listings” on page 69.

The FLEX-ES configuration files (resources and systems) must be *compiled*, using the command **cfcomp**. Compilation produces two files. Parts of the output file names are taken from the names in the definition files and part of the names (syscf and rescf) are fixed. In our case we will have S10A.syscf and R10A.rescf, based on the names coded in our definitions.

Once compiled, the resource portion (the rescf file) can be activated with the **resadm** command:

```
$ cd /usr/flexes/rundir
$ su                                        (switch to root)
# resadm -s R10A.rescf                      (must be root to use -s)
# exit                                      (switch back to flexes)
```

You can list all active resources with the command:

```
$ resadm -r
Resource: CPU Flags: READY Type: CPU Port: 9365 Pid: 17483
Resource: CHANNEL Flags: READY Type: CHAN Port: 9369 Pid: 17485
Resource: os2821 Flags: READY Type: CU Port: 9368 Pid: 17486
Resource: os3480 Flags: READY Type: CU Port: 9370 Pid: 17487
Resource: os3274 Flags: READY Type: CU Port: 9374 Pid: 17488
Resource: osdasd Flags: READY Type: CU Port: 9375 Pid: 17489
Resource: os3172 Flags: READY Type: CU Port: 9377 Pid: 17490
Resource: osctc Flags: READY Type: CU Port: 9376 Pid: 17491
Resource: NETCU Flags: READY Type: NETCU Port: 9384 Pid: 17492
Resource: TS3270 Flags: READY Type: TS3270 Port: 9381 Pid: 17493
```

More options of the **resadm** command are discussed in “The resadm command” on page 81.

Once the resources are active, the *system* portion is ready to be started and IPLed. This is easier to do with a shell script.

### 4.1.1 Building a shell script

We used **vi** to create the following shell script file in /usr/flexes/rundir/shos. (The name *shos* is arbitrary.) The contents are:

```
flexes S10A.syscf
xmodmap -e ‘keysym Alt_L = Alt_L Meta_L‘
xset fp+ /usr/flexes/fonts
xset fp rehash
echo '  Verify your master console session is started '
echo '  and then ipl your system '
echo '  '
x3270 -model 3 -keymap pc -port tn3270 localhost:mstcon &
x3270 -model 3 -keymap pc -port tn3270 localhost:L701 &
flexescli localhost S10A
```

The **flexes** command (in the shell script) starts an emulated S/390, using the indicated compiled system definition file.

The next three commands prepare `X windows` parameters, and the **x3270** commands start 3270 client windows on the Linux screen. We elected to identify specific 3270 sessions (mstcon and L701) for these two 3270 client windows.[2] These session names match names in our resources definition. There is no requirement to start these x3270 sessions here (in the shell script), but the terminal for the OS/390 master console(s) should be started before IPLing OS/390. Using the shell script this way provides a convenient, automatic way to establish the OS/390 console.[3]

The last command of this shell script, **flexescli localhost S10A**, starts the CLI program in interactive mode, with a *flexes>* prompt replacing the default Linux prompt. You can enter *flexes* commands here,[4] or enter a **quit** command to return the Window session to a Linux prompt. In most cases, you will not need this Window session while you are using OS/390.

Assuming that we have already started the resource manager (with a **resadm -s S10A.rescf** command) we can now invoke the shell script to start S/390 operation:

```
$ sh shos
flexes> ipl a80 0a82cs
flexes>
```

Here we started the **shos** shell script. The last command in the shell script starts **flexescli** (the command line interpreter, or CLI) in interactive mode, and this program provides the *flexes>* prompt. At this point we can enter CLI commands, some of which are outlined in “CLI commands” on page 82. We entered an **ipl** command to start OS/390. We could have included the **ipl** command in the shos script, but we found it more convenient to enter it manually because we often change the IPL parameter values. (The “0a82cs” in the example is an IPL parameter value.)

If we wanted to include an **ipl** command in the shell script, it could look like this:

```
flexes S10A.syscf
xmodmap -e ‘keysym Alt_L = Alt_L Meta_L‘
xset fp+ /usr/flexes/fonts
xset fp rehash
x3270 -model 3 -keymap pc -port tn3270 localhost:mstcon &
x3270 -model 3 -keymap pc -port tn3270 localhost:L701 &
echo ‘ipl a80 0a82cs’ | flexescli localhost S10A
flexescli localhost S10A
```

This illustrates the two ways in which **flexescli** can be used. If a command is piped to it (with **echo**), then **flexescli** will execute that command and quit. If nothing is piped to it (as in the last line of the shell script), then **flexescli** starts in interactive mode and issue the *flexes* prompt. The two operands for **flexescli** are the IP name/address of the system running FLEX-ES (localhost) and the name of the FLEX-ES instance (S10A). The requirement for these two operands becomes more apparent if you are running multiple FLEX-ES images with networked channels.

Note that we hard-coded the name of our FLEX-ES instance (S10A) in the shell script. If we intended to use several different FLEX-ES defined systems, we would need multiple shell scripts.

---

[2] If we omitted the session identification, we would obtain the Terminal Solicitor selection menu on these sessions.

[3] If you do not start the x3270 sessions this way, you would need access the Terminal Solicitor and start the TN3270 session that will be used for the OS/390 console before starting an IPL.

[4] For example, you could enter an IPL command at this point. Some users prefer to code an IPL command in their shell script, while others prefer to enter it at the *flexes* prompt.

### Terminal Solicitor

Starting the resource manager, **resadm**, automatically starts the FLEX-ES Terminal Solicitor. In order to connect to the FLEX-ES Terminal Solicitor, we connect a TN3270 client to the Linux IP address using port 24. Port 24 is the default port for the Terminal Solicitor. The client system can be connected to any LAN that is connected to our Linux[5]. The Terminal Solicitor presents a panel to us with the names of *available*[6] 3270 devices (these are often made available by the FLEX-ES CLI **mount** command, although we elected to hard-code them in the resource definitions):

```
Welcome to the FLEX-ES Terminal Solicitor (node: itsoefs1)

Please select (X) the desired service and press enter
(PA1 to exit; CLEAR to refresh)

_ L702          _ L703
_ L704          _ L705          _ L706          _ L707
_ L708          _ L709          _ L70A          _ L70B
_ L70C          _ L70D          _ L70E          _ L70F
```

*Figure 4-1 Terminal Solicitor panel (on a TN3270 screen)*

If we had not started the Linux x3270 sessions for *mstcon* and *L701* in our shell script, then these names would also appear in the Terminal Solicitor list. We would need to select *mstcon* from the list to activate that session before IPLing OS/390, since it will be used for the MVS master console.[7] When we select a terminal (by marking it with an X and pressing Enter[8]), the Terminal Solicitor screen (on our client TN3270 session) is replaced by a 3270 connection to the indicated terminal address. After OS/390 is up, this is likely to result in the USSTAB logo screen.

## 4.1.2 IPL OS/390

The OS/390 IPL starts when you execute a *flexes* **ipl** command. The **ipl** command can be entered from the *flexes* window or in a startup script, as shown in 4.1.1, “Building a shell script” on page 29.[9] Once our OS/390 was operational, we tried a few typical OS/390 commands, as shown in the following (the operator commands are shown in bold type):

```
- 11.06.30          d m=stor
   11.06.32           IEE174I 11.06.30 DISPLAY M 454
   REAL STORAGE STATUS
   ONLINE-NOT RECONFIGURABLE
       OM-256M
   ONLINE-RECONFIGURABLE
       NONE
   PENDING OFFLINE
       NONE
```

[5] It can also be another window on our Linux desktop.

[6] Once a 3270 terminal device is selected by a user, it is removed from the Terminal Solicitor panel. When the device is freed by the user, it reappears on the Terminal Solicitor panel and can be selected by another user.

[7] The AD system expects a 3270 terminal at address 700 for its master console. The name *mstcon* is arbitrary, and matches the name chosen for a 3270 defined at address 700 when the resource definition file was created.

[8] Many 3270 emulator users set up the right-hand Ctrl key as the logical 3270 Enter key, since this most closely matches a “real” 3270 keyboard. We did this and, using the IBM PCOM emulator, we pressed the right-hand Ctrl key.

[9] We eventually removed it from our shos script because we frequently wanted to IPL with various different IPL parameters. We used the same shos script (less the **ipl** command) and simply typed our **ipl** command at the *flexes* prompt.

```
- 11.42.47          d m=cpu
  11.42.47          IEE174I 11.42.47 DISPLAY M 662
  PROCESSOR STATUS
  ID  CPU                 SERIAL
  0    +                  00F0001247

  + ONLINE    - OFFLINE    . DOES NOT EXIST  W WLM-MANAGED

- 11.43.44          d a,l
  11.43.44          IEE114I 11.43.44 2000.314 ACTIVITY 665
   JOBS     M/S    TS USERS    SYSAS    INITS   ACTIVE/MAX VTAM     OAS
  00002    00011    00001      00027    00012    00001/00010       00007
   JES2     JES2     IEFPROC  NSW  S  VTAM     VTAM     VTAM     NSW  S
   DLF      DLF      DLF      NSW  S  RACF     RACF     RACF     NSW  S
   VLF      VLF      VLF      NSW  S  INETD4   STEP1    BPXOINIT OWT  AO
   LLA      LLA      LLA      NSW  S  TSO      TSO      STEP1    OWT  S
   SDSF     SDSF     SDSF     NSW  S  TCPIP    TCPIP    TCPIP    NSW  SO
   PORTMAP  PORTMAP  PMAP     OWT  SO FTPD1    STEP1    FTPD     OWT  AO
   NFSS     NFSS     GFSAMAIN NSW  SO
  P390    OWT

- 11.41.55          d u,,,a80,8
  11.41.55          IEE457I 11.41.55 UNIT STATUS 659
  UNIT TYPE STATUS        VOLSER     VOLSTATE
  0A80 3390 S             OS39RA     PRIV/RSDNT
  0A81 3390 A             OS3RAA     PRIV/RSDNT
  0A82 3390 A             OS39M1     PRIV/RSDNT
  0A83 3390 F-NRD                        /RSDNT
  0A84 3390 F-NRD                        /RSDNT
  0A85 3390 F-NRD                        /RSDNT
  0A86 3390 F-NRD                        /RSDNT
  0A87 3390 A             OS39HA     PRIV/RSDNT
```

The processor type (in the **d m=cpu** response) is 1245. This is the processor type for a Netfinity/EFS system. Otherwise, the displayed responses are exactly the same (except, perhaps, for storage sizes) as would be found when running on any S/390 hardware.

## 4.2 Operation and use

Once you have OS/390 IPLed, operation is the same as for any other OS/390 installation.

### 4.2.1 User terminal connection

A TSO user, for example, would normally connect to the system by connecting a TN3270 session to the Linux IP address (or name, if you have a Domain Name Server service), port 24. This should produce the Terminal Solicitor display, as shown in Figure 4-1 on page 31. The user would then select one of the available sessions (in the Terminal Solicitor display) and would be connected to this S/390 address. When he eventually drops the TN3270 session, the S/390 3270 address will be restored to the Terminal Solicitor display for others to use.

If you start OS/390 TCP/IP, a user could connect directly to it by using TN3270 and the IP address you assigned to OS/390 TCP/IP. In this case, the user would not be connected to the Terminal Solicitor, but would be connected directly to OS/390 TCP/IP.

### 4.2.2 Linux TN3270

A TN3270 emulator is included with Linux. In our opinion, it is not quite the same quality as the x3270 emulator distributed with FLEX-ES or the PCOM emulator we used on OS/2 and Windows machines.

### 4.2.3 IODF requirements

OS/390 requires an IODF data set that defines the I/O configuration seen by the software. This normally matches the IOCDS defined for the S/390 I/O hardware configuration. The ThinkPad/EFS platform does not have an IOCDS. All resources are defined in FLEX-ES *system* and *resource* files, compiled with the FLEX-ES resource compiler **cfcomp,** and then activated by the FLEX-ES resource manager **resadm** command.

An IODF is still required within an OS/390 system, but the HCD input to generate it does not need to define control unit details. That is, a simple device definition (device number, type, optional features) is all that is required. OS/390 dynamic I/O redefinition capability is not available.

In response to a **d ios,config(all)** command, we received the following:

```
- 11.38.06           d ios,config(all)
  11.38.06           IOS506I 11.38.06 I/O CONFIG DATA 656
  ACTIVE IODF DATA SET = SYS1.IODF01
  CONFIGURATION ID = CBIPO          EDT ID = 00
  HARDWARE SYSTEM AREA DATA COULD NOT BE OBTAINED
  ELIGIBLE DEVICE TABLE LATCH COUNTS
          0 OUTSTANDING BINDS ON PRIMARY EDT
```

You can, however, perform *software* dynamic configuration changes via OS/390 Hardware Configuration Definition (HCD) dialogs or the OS/390 ACTIVATE command, provided the affected devices are included in the FLEX-ES definition files.

### 4.2.4 System performance monitors

Because I/O sub-channel blocks are not maintained by FLEX-ES software emulation, the OS/390 Resource Measurement Facility (RMF) is not fully supported. You can run RMF, or other system performance monitors, but some of the reporting (especially when it comes to I/O activity) will not be complete.

When we started RMF, it reported the absence of an IOCDS in this environment and automatically terminated I/O queuing activity reporting:

```
- 11.22.34           s rmf
- 11.22.35 STC00439  $HASP373 RMF      STARTED
  11.22.36 STC00439  ERB100I RMF: ACTIVE
  11.22.36 STC00439  ERB265I RMF: IOCDS INFORMATION UNAVAILABLE TO RMF.
   RESPONSE CODE 01F0
  11.22.37 STC00439  ERB260I ZZ : I/O QUEUING ACTIVITY RMF REPORT TERMINATED
  11.22.38 STC00439  ERB100I ZZ : ACTIVE
```

SYS1.LOGREC may not have hardware error information. Consequently, any report produced by Environmental Recording and Editing Program (EREP) will have limited value when it comes to hardware detected errors. Due to time constraints, we did not investigate this area in more detail.

### 4.2.5 TCP/IP for OS/390

It is possible to share a single Ethernet LAN adapter in the ThinkPad between Linux (including the FLEX-ES Terminal Solicitor) and OS/390 TCP/IP. This requires some coordination between the FLEX-ES definitions and the OS/390 TCPIP profile data set parameters. Our goal was to assign IP address 9.12.17.211 to OS/390 TCP/IP, using the OS/390 device addresses E20 and E21 for the interface. (Our Linux IP address was 9.12.17.210.)

We included the following FLEX-ES definitions:

```
System definition:

   channel (3) local
   cu devad(0xE20,2) path(3) resource(R10A3172)

Resource definitions:

   R10A3172: cu 3172
   interface local(1)
   options 'ipaddress=9.12.17.211'
   device(00) 3172 eth0
   device(01) OFFLINE
   end R10A3172
```

We changed the following lines in the TCPIP.PROFILE.TCPIP data set:

```
...
DEVICE LCS1 LCS E20
LINK ETH1 ETHERNET 0 LCS1
...
HOME 9.12.17.211 ETH1
...
GATEWAY
9    =     ETH1     1492     0.255.255.0    0.12.17.0
...
START LCS1
```

This specifies the use of LAN adapter 0 (sometimes known as the MPTS adapter number, from the OS/2 **MPTS** program). Adapter 0 is the default LAN adapter number for FLEX-ES. This number is not related to any Linux LAN number.

In order to share an Ethernet adapter with Linux, we must specify, in the resources file, the IP address to be used by OS/390. It must also be specified in the OS/390 TCP/IP profile data set, as shown here. With the specifications described here, we have these connection options from another host on the network:

- **TN3270** to 9.12.17.210, port 24 -- this connects to the Terminal Solicitor (assuming **resadm** is started. We can then select a session and should receive the USSTAB logo screen.
- **TN3270** to 9.12.17.211, port 23 -- this connects directly to OS/390 TCP/IP. In recent AD systems (where a USSTAB function is defined for TCP/IP) this should produce the logo screen.
- **telnet** to 9.12.17.210, port 23 -- this connects to Linux (if telnet connections are enabled).
- **telnet** to 9.12.17.211, port 1023 -- this connects to OS/390 Unix System Services, as an ASCII terminal (assuming you are using a recent AD system).
- **telnet** to 9.12.17.211, port 23 -- this produces a line-mode connection (via VTAM) to TSO and is not very useful.

## 4.3 Shutting down

Shutting down the system involves the following steps:

- Shut down OS/390. (With the AD systems we would enter **s shutdown** on the MVS console to start a VTAMAPPL script that issues various commands.)
- Using the *flexes* window:

```
flexes> shutdown                    (terminates S/390 emulation)
$ su                                (must be root to terminate resadm)
$ password:
# resadm -T                         (terminate all resource managers)
# resadm -k                         (end resadm)
# exit                              (leave root)
$
```

To start the system up again, we would use the **resadm -s xxx.rescf** command (using the name of our compiled resource file), start the shell script, and then issue an **ipl** command from the *flexes* prompt.

5

# Additional Topics

Most of this redbook is concerned with the installation of a basic, entry-level ThinkPad/EFS system. Many additional options and techniques can be used to enhance the entry-level system, and some of these and other topics are briefly discussed here.

© Copyright IBM Corp. 2001

## 5.1 Basic debugging

We present here a few common debugging steps that may be useful if you have a problem starting FLEX-ES. We must note that FLEX-ES typically does not produce error messages oriented to a S/390 person. An error message indicates that an error occured, but you may need to ignore the specific text of the message and attempt to deduce the original cause of the problem.

A number of common errors occur if userid *flexes* and groupid *flexes* do not own the emulated I/O files and most of the files in /usr/flexes/rundir. (We assume your *rescf* and *syscf* files, along with any shell scripts you might use, are in this directory.) You can use a command such as **`ls -al /usr/flexes/rundir`** to verify ownership, and a command such as **`chown flexes:flexes /usr/flexes/rundir/shos`** to change ownership. (Of course, if you **`cd`** to the relevant directory first, you will not need to enter full path names to files in that directory.)

Do all your resources start when you use the **`resadm -s xxx.rescf`** command? You can check this with a **`resadm -r`** command to list the currently active resources. You might verify that the resource names (listed by the **`resadm -r`** command) exactly match the resource names you specified in your system definition.

Do your *rescf* and *syscf* files match your current definitions (in your definition source file)? Changing your definition source file(s) has no effect until you recompile them and start (or restart) **`resadm`** with the newly compiled files. If you are not certain, you can simply compile your source file again. In our examples, this would be:

```
$ cd /usr/flexes/rundir
$ cfcomp defA1                          (using your correct source file name)
```

Be certain you have a clean compile. This is indicated by the messages:

```
Configuration processing *SUCCEEDED* with no errors
Data Space Manager Terminated
```

Are you using the correct files? The **`resadm`** command uses the *rescf* file and the **`flexes`** command uses the *syscf* file. It is easy to reverse the two file suffixes, and the resulting error message does not identify the exact problem.

Once the resource manager (**`resadm`**), the emulator (**`flexes`**), and the command line interface (**`flexescli`**) are started, you may find the following CLI commands useful:

```
flexes> d devstate 700                  (display state of emulated device)
flexes> d ckdcachestats A80             (verify I/O occurs to a disk)
flexes> d psw
```

If your emulated 3270 terminal operation is erratic, be certain you know where the 3270 Enter key is located, and where the 3270 New Line key is located. You may be creating a 3270 Enter operation when you intended to use the New Line key.

Be certain that your the disk model numbers are consistent. For example, if you created a 3390-2 (using **`ckdfmt`** or **`ckdconvaws`**) but you specify a 3390-3 in your resource definitions, the system will not start.

## 5.2 Operating Systems Messages console

Larger S/390 machines have an Operating Systems Messages console function that is provided through the Support Element (SE) or a Hardware Management Console (HMC). This console function is sometimes known as the “system console” or the “hardware system console.” OS/390 attempts to use it if all other MVS consoles fail.

FLEX-ES emulates this console through the CLI window--the window with the *flexes* prompt. Messages written from the S/390 to the Operating Systems Messages console appear after the *flexes* prompt. You need to press Enter (with the desktop focus in this window) to restore the *flexes* prompt. You can reply or enter commands through the Operating System Messages console by using the CLI command **`hwc`**:

```
flexes> SPECIFY SYSTEM PARAMETERS                (message from OS/390)
                                                 (press Enter to get flexes prompt)
flexes> hwc R 00,CLPA
flexes>
```

## 5.3 Security

As we explained earlier, FLEX-ES is a layer of software that resides and operates between an OS/390 system and an underlying Linux system. All the security features and functions that come with an OS/390 system work as on any other S/390 platform. However, it is possible for a Linux user with sufficient privilege to gain access to the contents of an emulated DASD or central storage associated with an emulated CPU, and so forth.

A ThinkPad/EFS owner must plan and manage traditional Linux security functions for the underlying Linux system, as well as traditional S/390 security management. If the ThinkPad/EFS platform is used only for S/390 operation, this can be fairly simple.

A unique concern involves the OS/390 master console(s). These can be implemented through the Terminal Solicitor. This is convenient, but offers an opportunity for an unwanted person to connect as a master console. We suggest that you have a master console on the ThinkPad display and direct any other master consoles[1] to specific IP addresses and not through the Terminal Solicitor. This is done by specifying an IP address in the FLEX-ES resources file, instead of a terminal name. This means, of course, that the client systems connecting to the emulated 3270 interfaces must have static IP addresses.

The FLEX-ES resource manager uses TCP/IP port 555 to talk with other instances of FLEX-ES resource managers. There may be a potential for problems if someone hacks this port. We are not aware of any instances of this, but it is a potential concern if you use an open network to link multiple FLEX-ES systems at this level. (Would you use an open network for shared DASD data flow on a “real” S/390? You need to look at this potential exposure from this viewpoint and not from a PC networking viewpoint.)

## 5.4 Server memory

The `memsize + essize + cachesize*11 + DASD cache` (described in “System definitions” on page 73 ) total values (when translated to bytes of storage) approximate the amount of Linux virtual storage needed to run an instance of S/390 emulation. If you emulate two S/390 systems (at the same time), you will need to add the values for each of the two emulated systems.[2] You can emulate more S/390 instances, but each one will require more memory.

[1] We are using the term “master console” loosely here to mean any OS/390 operator console.
[2] Some of the DASD cache memory may be shared among multiple emulated S/390s.

The key principle is that the Linux system *should not be forced to page* when running S/390 emulation. Linux might perform considerable paging when getting started, but once a S/390 instance is started and the system *working set* is established, the average Linux paging rate should be close to zero. (S/390 operating systems running in the emulated S/390 might have much higher paging rates; this is not the point at issue here.) *This almost always means that the ThinkPad real memory must be considerably larger than the emulated real memory of all the S/390 instances that are active at any one time.*

In principle, the Linux memory needed to emulate a S/390 is "just" virtual memory to Linux and might exist in a much smaller real memory used by Linux. In practice, this does not work well. Your ThinkPad should have enough memory so that all the S/390 requirements (memsize + essize + cachesize*11 + DASD cache) fit in your *real* ThinkPad memory, without requiring Linux paging. If you assume 100 MB for Linux, FLEX-ES programs, TCP/IP operation, and so forth, then the 512 MB ThinkPad in our examples might be used as follows:

```
Rough guess for Linux, FLEX-ES, etc              100 MB
S/390 memsize                                    256 MB
S/390 essize                                      64 MB
FLEX-ES instruction cache * 11                    22 MB
default DASD cache for 10 volumes (approx)        10 MB
                                                 --------
                                                 462  MB
```

This leaves a reasonable margin in a 512 MB system. The 100 MB for Linux is just a guess. The *working sets* (as seen by Linux) of the emulated S/390 memory (central and expanded), the FLEX-ES instruction cache, and most of the DASD cache will normally be their full sizes. The working set of Linux itself and its many system processes are much harder to determine.

## 5.4.1 The vmstat command

The traditional UNIX command for monitoring swapping[3] rates, **sar**, is not available in base Linux distributions. The **vmstat** command can be used instead. For example, the command **vmstat 10 2** means to run **vmstat** with 10 seconds between reports and quit after 2 reports.

```
# vmstat 10 2
  procs                  memory    swap          io      system           cpu
r  b  w   swpd   free  buff  cache  si  so    bi    bo    in    cs  us  sy   id
0  0  0      0 329156 28768  70156   0   0     4     1   105   196   1   0   99
0  0  0      0 329156 28768  70156   0   0     0     0   105   196   1   0   99
```

You should refer to your Linux documentation or **man** listings for more complete information about **vmstat**. Very briefly, the key fields are documented as:

- Procs
  - r: number of processes waiting for CPU time
  - b: number of processes in uninterruptable sleep
  - w: number of processes swapped out but otherwise runnable
- Memory
  - swpd: the amount of virtual memory used (in the swap file) (kB)
  - free: the amount of idle memory (kB)
  - buff: the amount of memory used as buffers (kB)
- Swap
  - si: amount of memory swapped in from disk (kB/s)
  - so: amount of memory swapped out to disk (kB/s)
- IO
  - bi: blocks sent received from a block device (blocks/s)

[3] In S/390 terms, we would say *paging* rates. In older contexts, *swapping* has a different meaning than *paging*. In modern Linux systems, *swapping* appears to have exactly the same meaning as *paging*.

  - bo: blocks send to a block device (blocks/s)
- System
  - in: the number of interrupts per second (including clock interrupts)
  - cs: number of context switches per second
- CPU
  - us: user time (percent of CPU time)
  - sy: system time (percent of CPU time)
  - id: idle time (percent of CPU time)

We are particularly interested in swap (paging) rates. Notice that the unit of measurement is kilobytes per second.

The Linux swapping rate is meaningful (for our discussion) only in a steady-state condition with a typical S/390 workload. Linux swapping while booting, or while starting FLEX-ES, is not relevant. Unusual S/390 work, such as CLPA processing or very unusual disk access patterns (affecting disk caches) might temporarily drive Linux into swapping. This is not good, but can probably be tolerated for short periods.

## 5.4.2 Importance of Linux swapping

Why is the Linux swapping rate so important? A reasonable analogy is CICS paging in an OS/390 system. A system with many TSO users might have sustained paging rates of hundreds of pages per second (on a larger S/390) with no ill effects, but CICS on the same system would require a paging rate close to zero. The problem is that the whole address space (CICS, for example) is placed in *wait* when a page fault occurs. Placing CICS[4] in *wait* causes all the CICS users serviced by that address space to *wait* while the page fault is resolved. A page fault in a TSO user address space causes only that one user to wait.

FLEX-ES operation is close to the CICS analogy. A Linux page fault in a key FLEX-ES process may cause the whole emulated S/390 instance to wait until the Linux page fault is resolved.

Remember that a page fault in a FLEX-ES S/390 instance has a very different effect than a page fault in Linux. For one thing, the S/390 page fault is seen only by the FLEX-ES emulation program--it is an emulated page fault. It is handled, by OS/390, as a S/390 page fault. If it occurs in a TSO user address space or a batch address space, it affects only that address space. If it occurs in a CICS address space, it affects all the users of that CICS. This is business as usual for OS/390.

The key message is that you should adjust your FLEX-ES system parameters (emulated S/390 memory, disk caches, instruction cache) to avoid Linux swapping. Defining a smaller emulated S/390 memory size may increase OS/390 paging. Of course, it would be nice to avoid any paging, but OS/390 paging is much less damaging than Linux swapping and your tradeoffs should always be in this direction. You can juggle disk cache versus instruction cache versus S/390 memory allocations for your best performance. Simply be careful not to push Linux into swapping.[5]

[4] Modern CICS systems ameliorate this situation in various ways; the description here should be regarded as conceptual.

[5] Again, we stress that “Linux paging” refers to steady-state operation after S/390 emulation is started. Linux booting or FLEX-ES startup may cause Linux paging and we are not concerned with this temporary effect.

## 5.5 Ultrabay

ThinkPad A21 and A22 units have a “bay” that can hold a CD-ROM drive, or a diskette drive, or a second hard disk drive—but only one at a time. Current Windows operating systems permit the user to change these devices while the system is running. We were unable to make clean (without side effects) changes under Linux, while Linux was running. We found it was easier to shut down Linux, let the ThinkPad power down, and then swap devices in the Ultrabay. Linux detected the new device correctly when it was restarted.

### 5.5.1 Using a second Linux hard disk

The preparation and use of a second hard disk is discussed in “Ultrabay” on page 42. After it was available, we created three 3390-1 work volumes on the second hard disk. The following FLEX-ES commands, issued from a Linux Terminal window, will do this:

The **ckdfmt** commands take some time to run because both write the complete emulated 3390 volume. The **-n** flag in **ckdfmt** prevents a verification pass to read the formatted data just written. These commands do not perform S/390 operating system initialization. After OS/390 is running, we will need to use ICKDSF to initialize the volumes.

We obtained a second 32 GB hard disk drive (IBM part number 08K9511), along with the mounting tray (IBM part number 08K6068) needed to use it in the Ultrabay. We removed the CD-ROM drive and installed the second hard disk while the ThinkPad was turned off. We turned power on and let Linux boot. We then determined the Linux identity of the second drive:

```
# cat /proc/partitions
   major   minor   #blocks   name   (.... more data .....)
     3        0              hda
     3        1      nnnnn   hda1
     3        2      nnnnn   hda2
    ..       ..       ....    ....
    22        0    31253040   hdc    0 0 0 0 0 0 0 0 0 0 0
```

The first hard disk (internal in the ThinkPad) is /dev/hda; looking at the above listing, we see a second hard disk as /dev/hdc. This drive (as expected for a new disk) had no partitions. (We do not know why Linux selected *hdc* instead of *hdb*, which would logically be next in sequence.)

We decided to create two file systems on the second hard disk. One would be about 2 GB and use mount point */holding*. The other would use all the remaining space and have mount point */s391*. We intended to use */holding* for compressed tar images (created as backups) and other temporary files and we might have considerable creation/deletion activity in this file system. The */s391* file system would be solely for emulated S/390 volumes. Using two file systems seemed to reduce the chances for significant fragmentation of emulated volume files.[6]

We also noted that IBM, like most other manufacturers, uses decimal numbers to describe disk capacity (1M = 1,000,000) while Linux utilities typically use power-of-two numbers (1M = 1,048,576). Using Linux numbers, the capacity of the disk was about 29 GB.

[6] We may have been too sensitive about fragmentation. Experienced Linux users claim there are practically no fragmentation effects in normal Linux file systems. However, they admit that FLEX-ES operation (57 KB reads or writes when emulating a 3390) is not typical Linux I/O. We felt that disk I/O may be the weakest point of ThinkPad/EFS performance and felt that avoiding fragmentation might help performance. The author welcomes any measurement data (on an EFS system) that may help resolve this issue.

We partitioned the disk with **fdisk** and formatted it:

```
# fdisk /dev/hdc
  n                                 (fdisk option to add a partition)
  Command action: p                 (a primary partition)
  Partition number: 1
  First cylinder (1-66144): 1       (default = 1)
  Last cylinder (1-66144): 4140     (this makes a partition of about 2 GB)
  n                                 (fdisk option to add a partition)
  Command action: p                 (another primary partition)
  First cylinder (4141-66144): 4141
  Last cylinder (4141-66144): 66144
  p                                 (print the partition table)
     Device Boot    Start   End    Blocks       ID   System
      /dev/hdc1      1     4140    1956118+     83   Linux native
      /dev/hdc2     1414  66144    29296880     83   Linux native
(if the ID is not 83 = Linux native, use the t option to change it)
  t                                 (fdisk option to change partition ID)
  Partition number: 1
  Hex code: 83                      (83 is Linux native)
  t
  Partition number: 2
  Hex code: 83
  p                                 (fdisk option to list partitions)
  w                                 (fdisk option to write partition table and end)
# mkdir /s391                       (create a mount point, if not already done)
# mkdir /holding
# mke2fs /dev/hdc1                  (format a file system)
# mke2fs /dev/hdc2                  (format the other file system)
# mount /dev/hdc1 /holding          (mount it for Linux use)
# mount /dev/hdc2 /s391
# chown flexes:flexes /holding      (change owner and group identities)
# chown flexes:flexes /s391
```

We used the new disk space to confirm that it worked, and then edited */etc/fstab* to cause the new file systems to be automatically mounted at boot time. (There are no unwanted effects if the second hard disk is not present when Linux is booted; the file systems on that disk are simply not mounted.) We added two lines at the end of fstab:

```
LABEL=/         /            ext2   defaults        1 1
LABEL=/boot     /boot        ext2   defaults        1 2
/dev/fd0        /mnt/floppy  auto   noauto,owner    0 0
LABEL=/s390     /s390        ext2   defaults        0 0
none            /proc        proc   defaults        0 0
none            /dev/pts     devpts gid=5,mode=620  0 0
/dev/hda7       swap         swap   defaults        0 0
/dev/hdc1       /holding     ext2   defaults        0 0
/dev/hdc2       /s391        ext2   defaults        0 0
```

## 5.5.2 Second disk planning

It is unlikely that the second hard drive will *always* be installed when FLEX-ES is running. Although OS/390 does not use the CD-ROM drive or the diskette drive (the other possible occupants of the Ultrabay), you may find it inconvenient to shut down the ThinkPad to remove the CD-ROM drive or diskette drive and insert the second hard disk drive every time you want to run FLEX-ES.

The difficulty is that, if a file on the second hard disk (such as an emulated S/390 volume) is specified in a FLEX-ES resources definition, then that file *must* be available before **resadm** can be started. Linux mounting (through /etc/fstab) forgives missing file systems, but **resadm** does not. We handled the situation this way:

- We defined a complete IPLable system on the primary hard disk drive. (For the OS/390 R10 AD system, this required four 3390-3 drives.)
- We defined a FLEX-ES system and resource definition that referenced only these drives. These are the definitions (S10A and R10A) that we have described earlier. If only one hard disk drive is installed, we can start FLEX-ES using these definitions.
- We placed optional OS/390 volumes (DLIBs, work volumes, and so forth) on the second hard disk drive.
- We created another set of FLEX-ES definitions (S10B and R10B) that specified the additional S/390 volumes on the second hard disk drive. We created a second shell script (shosb) that uses this system name. If both hard disks are installed, we start FLEX-ES with these definitions.

There is a slight inconvenience in this method. We need to keep the common parts of both definitions synchronized, and this sometimes means double editing is required when changes are made. Both sets of definitions and the associated shell scripts are listed in Appendix A, “FLEX-ES definition listings” on page 69.

The remaining general discussion in this section assumes that both disk drives are mounted.

### 5.5.3 Disk layout (AD system)

Since we had two hard disks for our ThinkPad/EFS system, we decided to split the S/390 volumes between them. This might produce a small benefit in overlapped disk operation. We decided to place an IPLable system on the primary drive and optional volumes on the second drive. There was no special reason for this plan, but it seemed reasonable. We had ample space and could install many more 3390 volumes, if needed. We could have, for example, installed two AD releases as well as several user volumes.

We used the following layout for our OS/390 AD R10 system:

```
hda   Primary hard disk drive
   partition    mount     size         Use.........
   hda1         /boot     36 MB        Linux boot partition
   hda5         /         3 GB         Linux root, containing all of Linux, FLEX-ES
   hda6         swap      150 MB       Linux swap
   hda7         /s390     26 GB        File system for S/390 volumes
hdc   Second hard disk drive
   hdc1         /holding  2 GB         Target area for tar/compress, and other uses
   hdc2         /s391     27 GB        File system for S/390 volumes
```

We placed our S/390 volumes (for the OS/390 AD R10 system) as follows:

```
/S390     (on Primary hard disk)
   Device    Volser    Addr   Use
   3390-3    OS39RA    A80    IPL volume
   3390-3    OS3RAA    A81    More OS/390 libraries
   3390-3    OS39M1    A82    Parmlibs, spool, paging, VSAM, etc
   3390-3    OS39HA    A87    HFS data sets
   (14 GB still available)

/S391     (on Second hard disk)
   3390-3    OS39DA    A85    DLIBs
   3390-3    OS3DAA    A86    DLIBs
   3390-2    OS3DAB    A88    DLIBs
```

```
3390-2    OS39PA    A83    DB2
3390-2    OS3PAA    A84    Transaction Server (CICS)
3390-1    WORK01    A90    Work (storage) volume
3390-1    WORK02    A91    Work (storage) volume
3390-1    WORK03    A92    Work (storage) volume
(about 12 GB still available)
```

Since the CD-ROM drive and the second hard disk cannot be installed at the same time, it was necessary to stage volumes from the AD CD-ROMs. That is, we unzipped and converted (using **ckdconvaws**) OS39DA, OS3DAA, OS3DAB, OS39PA, and OS3PAA onto the first hard drive (mount point /s390).[7] The process of unzipping and installing a volume from an AD CD-ROM system is described in “Installation tasks” on page 24. We then shut down Linux, removed the CD-ROM drive and installed the second hard disk, and rebooted Linux. We then copied the appropriate files to the second disk (mount point /s391):

```
$ cp /s390/OS39DA /s391/OS39DA
$ cp /s390/OS3DAA /s391/OS3DAA
$ cp /s390/OS3DAB /s391/OS3DAB
$ cp /s390/OS39PA /s391/OS39PA
$ cp /s390/OS3PAA /s391/OS3PAA
$ rm /s390/OS39DA
$ rm /s390/OS3DAA
$ rm /s390/OS3DAB
$ rm /s390/OS39PA
$ rm /s390/OS3PAB
```

The three work volumes are not part of the AD system. We created them with the **ckdfmt** command:

```
$ ckdfmt -n /s391/WORK01 3390-1
$ ckdfmt -n /s391/WORK02 3390-1
$ ckdfmt -n /s391/WORK03 3390-1
```

### 5.5.4 Alternative method

We described using two FLEX-ES definition files as a way to deal with a second hard disk drive that is not always present. An alternative is to use a single FLEX-ES resources definition file that specifies all the emulated S/390 drives on the second hard disk as OFFLINE. Two startup shell scripts would be used. One (for use when both hard disks are present) would contain a series of FLEX-ES **mount** commands to, in effect, bring the volumes on the second hard disk online.

## 5.6 FLEX-ES FakeTape on OS/390

FakeTape[8] emulates tape devices using Linux disk files instead of tape drives. Provided the appropriate tape devices are defined in the OS/390 IODF configuration data set, FakeTape will emulate any type of tape drive from 3420 to 3490-E. Because FakeTape always writes and reads the same format to/from Linux, it operates at the same speed for all different emulated tape device types. We ran several tape jobs using IEBGENER, IEBCOPY and DFDSS and they all performed well. Our definitions included a 3480 tape drive at address 560.

This is one of the jobs we executed:

```
//P390T    JOB 1,OGDEN,MSGCLASS=X
```

[7] We elected not to restore the other AD volumes containing IMS and WebSphere components.
[8] FakeTape is a trademark of Fundamental Software, Incorporated.

```
//BACKUP   EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT1   DD DISP=SHR,DSN=SYS1.PROCLIB(JES2)
//SYSUT2   DD DSN=PROCLIB,DISP=(NEW,KEEP),
//            UNIT=560,LABEL=(1,SL),VOL=SER=222222
```

We submitted the job and the IEF233A mount request was issued by OS/390. The console interaction went like this:

```
   $HASP373  P390T STARTED - INIT 1
   IEF244I   P390T - UNABLE TO ALLOCATE 1 UNIT(S)
   IEF877E   P390T NEEDS 1 UNIT(S)
   FOR SYSUT2
   FOR VOLUME 222222
   OFFLINE
   0560
02 IEF238D   P390T - REPLY DEVICE NAME OR 'CANCEL'
   2,560

   IEF503I UNIT 0560 I/O ERROR
   IEF234E D 0560
   IEF233A M 0560,222222,P390T,,PROCLIB

(at this point, go to the flexescli window and issue the command
(flexes> mount 560 /tmp/222222

   IEC512I LBL ERR 0560,        ,NL,222222,SL,P390T.....
   IEC704A L 0560,222222,SL,NOCOMP,P390T......
03 IEC704A REPLY 'VOLSER,OWNER INFORMATION', 'M' OR 'U'
   3,222222
```

As seen in this interaction, we submitted our job. The tape drive was offline and OS/390 requested that we provide a device. We answered with the device address, 560. We next received a mount request for volume 222222 (taken from our JCL) on unit 560. At this point, we switched to the CLI window (with the *flexes* prompt) and entered **mount 560 /tmp/222222**. This file name was quite arbitrary; if you plan to use many emulated tape volumes, you will probably want to create a directory for them (and perhaps a whole file system for them).

OS/390 attempted to read the tape label (using an empty /tmp/222222 file) and received an error. This causes normal recovery processing for tape labels. We entered a volser (222222) in response to message IEC704A, and the job ran correctly. OS/390 rewound and unloaded the "tape" at the end of the job. This logically unmounted the file and made the tape drive logically not ready.

We later read the tape and printed (to SYSPRINT) the tape contents. The process was about the same, except our tape file (/tmp/222222) now had a correct label and was immediately accepted after we entered another *flexes*> **mount 560 /tmp/222222** command.

Server files used by FakeTape are normal files in the server's file systems. With a little planning—for space and naming—a whole tape library can be emulated using inexpensive server disks. A file (in the server file system) is equivalent to a tape volume. You can compress these files, ftp them, write them on CD-ROMs, and so forth, provided they are restored to their normal form when used again by FakeTape.

FakeTape can automatically recognize and handle data in AWSTAPE and AWSOMA formats, as produced by P/390-related systems. You should be able to exchange such files via ftp, although we did not try this. You can use the FLEX-ES utility **initawstape** to initialize a file in AWSTAPE format and then write output data to it.

Several special options are available for FakeTape resource definitions:

```
c3480: cu 3480
options 'maxwritesize=200,allowdisconnects,allowmountccws,autoloader'
device(00) 3480 OFFLINE
end c3480
```

The maxwritesize option is important. It specifies (in megabytes) the maximum size of an emulated output tape. If your output approaches this size, FakeTape will signal an end-of-tape reflective marker. The S/390 program would usually write trailer labels at this point. In any case, FakeTape will write additional records but signal end-of-tape for every additional write request. If you do not specify a maxwritesize value, two exposures occur:

- You might consume all the free space in the server file system.
- You will never have end-of-volume processing (triggered by an emulated end-of-reel reflective marker).

In our opinion, you should always have a maxwritesize specification when using FakeTape. Unfortunately, you cannot specify option values in a FLEX-ES **mount** command. You need to code a specific maxwritesize value in the resources definition and this will apply to any file (emulated tape volume) mounted on the device containing the maxwritesize parameter.

(The maxwritesize parameter also applies to DAT tapes. DAT drives do not signal that they are near the end of the tape (to permit trailer records to be written). They signal end-of-tape and nothing more can be written. A well-selected maxwritesize can overcome this limitation.)

The other options are more specialized and you may never need them with FakeTape.

## 5.7 Multi-system setup

You can define multiple S/390s by defining multiple FLEX-ES system definition files and compiling them. You can run multiple S/390 systems, one at a time, by simply creating multiple shell scripts (see "Building a shell script" on page 29) that name the appropriate *syscf* file in the **flexes** command. You can run multiple S/390 systems at the same time (*multiple instances*) if you have enough ThinkPad/EFS memory and have defined your FLEX-ES resources appropriately.

Only one resource definition (*rescf* file) can be active on a server, so it must define all the resources needed by all the S/390 instances. Each S/390 instance must be started with a unique system definition (*syscf* file). You would normally start a separate CLI window, with its *flexes* prompt, for each instance. Some care is needed to use the right *flexes* windows when controlling the S/390 instances. The CLI command **set prompt** can be used to change the *flexes* prompt to something more meaningful for each instance.

The redbook *NUMA-Q Enabled for S/390: Technical Introduction*, SG24-6215, describes the simultaneous operation of OS/390, VM/ESA, and VSE/ESA *instances*. A NUMA-Q system (now named x/Series 430 EFS) is larger than a ThinkPad/EFS system, but the details for running multiple FLEX-ES instances are the same. (One exception is that, with more Server processors, it is possible to use *dedicated* processors for *multiple instances* on the NUMA-Q machine, whereas this is not possible on a ThinkPad.)

## 5.8 x3270 client

The FLEX-ES product distribution includes a copy of x3270. This is based on the freely distributable x3270 package, but with a few modifications. In general, it is intended only for use on the ThinkPad/EFS system itself, under Linux. Users connecting from other platforms will probably use whatever TN3270 clients are available for their platforms. For this redbook project, we used IBM's PCOM package—usually from OS/2 machines.

An x3270 window (on the Linux console) is started with the command:

```
$ x3270 -model 3 -keymap pc -port tn3270 localhost &
    |       |         |          |          |      +-- run in background
    |       |         |          |          +--------- local system (from /etc/hosts)
    |       |         |          +-------------------- port tn3270 (from /etc/services)
    |       |         +------------------------------- in /usr/X/lib/app-defaults/X3270
    |       +----------------------------------------- models 2, 3, 4, 5 available
    +------------------------------------------------- command
```

You can optionally identify the FLEX-ES terminal session you want to connect to like this:

```
$ x3270 -model 3 -keymap pc -port tn3270 localhost:t91f &
```

In this case, your resources definition (or a prior FLEX-ES **mount** command) must have defined a 3270 device named *t91f*. If you do not identify a specific terminal connection, the x3270 session is connected to the Terminal Solicitor. In a FLEX-ES system, *port tn3270* normally resolves (via /etc/services) to port 24, the default port for the Terminal Solicitor.

The **x3270** command must be in the current search path, of course.[9] It is in the /usr/flexes/bin directory, and we suggested placing this in the current PATH. The *model* parameter refers to 3270 models:

```
Model     Characteristics
  2       24 lines x 80 columns
  3       32 lines x 80 columns
  4       43 lines x 80 columns
  5       27 lines x 132 columns
```

These are standard 3270 family model designations. Model 3 use is perhaps the most common and that is what we used.

Important keyboard mapping for x3270 includes:

```
Function     Keys Used
PA1          Left-Alt + 1          (hold down the left Alt key and press 1)
PA2          Left-Alt + 2
Clear        Left-Alt + c   or   Left-Alt + Home
Enter        Enter                 (large Enter key performs 3270 Enter function)
NewLine      Left-Alt + Enter      (performs 3270 New Line function)
SysRq        Left-Alt + s          (the System Request key (usually for SNA))
F13          Left-Alt + F1         (and so forth for F14 - F24)
EraseEOF     Left-Alt + f
Insert       Left-Alt + i
Del          Delete
BackTab      Shift + Tab           (tab backwards)
```

[9] You must make certain than another x3270 program does not appear first in the search path, as discussed in "X3270" on page 20.

This key mapping may be disconcerting to experienced 3270 users (especially ISPF users) because it uses the large Enter key (on the PC keyboard) as the 3270 Enter key. On a “real” 3270, the large key in this position performs a New Line function, and a smaller key (where the Right Cntrl key is positioned on a PC keyboard) performs the 3270 Enter function. You can correct this situation by the following:

```
# cd /usr/lib/X11/app-defaults
# cp X3270 X3270old                              (keep copy of original, just in case)
# vi X3270
.....
X3270.keymap.pc: \                               (scroll until you find this line)
   Meta<Key>d:         Redraw()\n\
   Alt<Key>d:          Redraw()\n\
   Meta<Key>Return:    Newline()\n\
   Alt<Key>Return:     Newline()\n\
   <Key>Return:        Newline()\n\              (change this line, as shown)
   !Shift<Key>Tab:     BackTab()\n\
   .....
   .....                                         (near end of 'X3270.keymap.pc' section)
   Alt<Key>Home:       Clear()\n\
   <Key>KP_Enter:      Enter()\n\                (insert this line)
   <Key>Control_R:     Enter()\n\                (insert this line)
   <Key>Control_L:     Reset()\n\                (insert this line)
   :<Key>:             Default()
X3270.keymap.pc84: \
.....
```

You will need to use *:w!* to save the file in **vi**, because it is a read-only file. Make these changes carefully; there is one altered line and three new lines. Be careful not to make any other changes unless you understand the format and functions of this file.

We changed the large Enter key on the PC keyboard to perform the 3270 New Line function, the Left Cntrl key to perform the 3270 Reset function, and the Right Cntrl key to perform the 3270 Enter function. This is the most familiar keyboard arrangement for most 3270 users.

## 5.9 Remote resources

FLEX-ES can use TCP/IP links to other instances of FLEX-ES on remote Servers. You could, for example, run only the FLEX-ES *resource manager* on a remote Server that has a large disk configuration. With the proper system definition (locally) and resources definitions (locally and remotely), a FLEX-ES instance could access the remote disks (as emulated S/390 devices) as readily as local disks. When used with a fast LAN, the performance is not much slower than local device access. Usage is not limited to emulated disks; for example, a SCSI tape drive on a remote system could be used, or emulated channel-to-channel connections. A FLEX-ES license for each system is required.

Setup for this type of operation can be recognized by the keyword *network* in channel definitions and in cu definitions. FLEX-ES automatically uses TCP/IP port 555 for communication between multiple *resource managers*.

We did not use remote resources for our redbook project. Our project concentrated on setting up and operating a basic ThinkPad/EFS system, and we did not need remote resources for this. However, FLEX-ES remote resources is potentially a very powerful and useful function in appropriate situations.

A conceptual network is shown in Figure 5-1. In this example, a single instance of FLEX-ES is executing (in system B in the center of the figure). It is using remote resources on the other two systems; disk resources on system A and a tape drive on system C. The FLEX-ES resource managers in all three systems communicate with each other. The emulated S/390 is unaware that some of the emulated devices are on remote systems.

System and device definitions are required on all three systems. These would include the following lines:

```
System A:
   adisk: cu 3990
   interface network(1)

System B:
   channel(3) network
   channel(4) network
   cu devad... path(3) resource(adisk)
   cu devad... path(4) resource(atape)

System C:
   atape: cu 3480
   interface network(1)
```

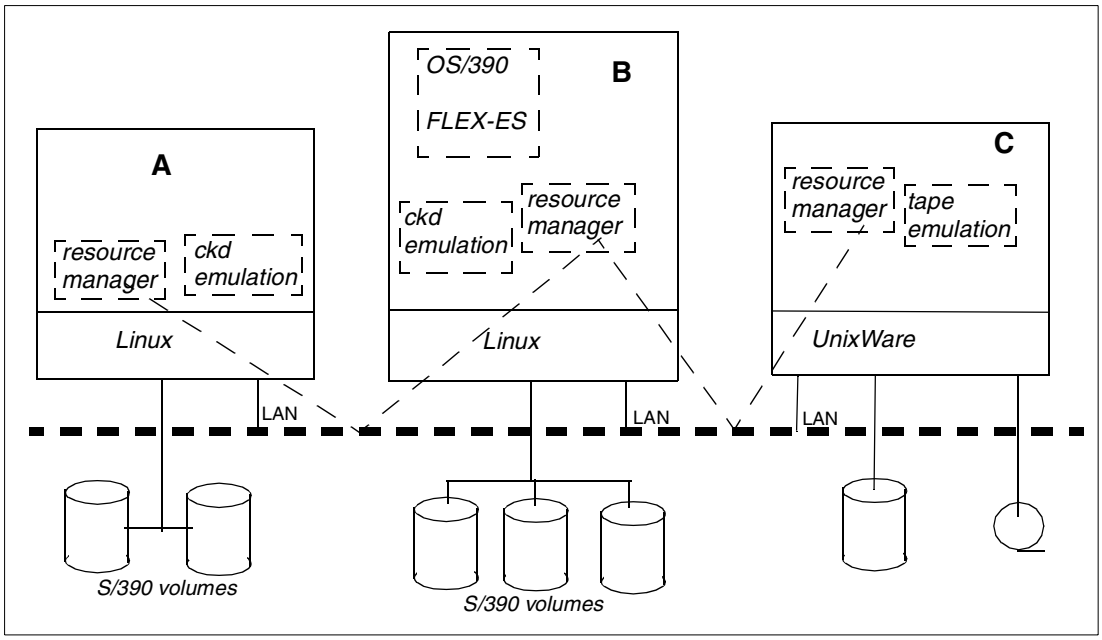


*Figure 5-1 FLEX-ES Networking*

## 5.10 Disk caches

FLEX-ES automatically caches 15 tracks of data for each emulated 3390 or 3380. You can adjust this in three ways:

- Specify a different number of tracks to cache for a particular emulated drive.
- Specify a different number of tracks to cache at the control unit level; excess tracks (above those needed for the specified or default cache for each device on the control unit) will *float*, as needed, among all the devices on the control unit.

- Use the *writethroughcache* parameter to force a different operation of the cache (on a device or control unit level). The default operation uses a *writeback* cache technique.

Here is an example that uses all three options:

```
(resources definitions)
....
c3990A: cu 3990
interface local(1)
options 'trackcachesize=150'
device(00) 3390-3 /usr/flexes/links/A3s1
device(01) 3390-3 /usr/flexes/links/B3s1 devopt 'trackcachesize=5'
device(02) 3390-3 /usr/flexes/links/C3s1 devopt 'trackcachesize=45'
device(03) 3390-3 /usr/flexes/links/D3s1 devopt 'trackcachesize=30,writethroughcache'
device(04) 3390-1 OFFLINE devopt 'trackcachesize=0'
end c3990A
```

This is a bit complex. The five devices defined will ask for (15 + 5 + 45 + 30 + 0 =) 95 tracks of cache. (Device (00) does not specify a cache size and defaults to 15 tracks.) The control unit definition specifies 150 tracks of cache. This is (150 - 95 =) 55 more tracks than needed by individual device caches, and the 55 tracks will be a *floating* cache. The floating cache is managed by internal FLEX-ES logic. Each 3390 track is about 57 KB, so the 150 tracks of cache will require about 8.3 MB of Server storage.

Cache is normally allocated for an offline device, since you might perform a FLEX-ES **mount** command to use the device. If you are certain you will not use the device (or you really want no cache for some reason), you can specify a cache of zero tracks.

If you specify a control unit cache size of less than the sum of the individual device caches, the specified control unit cache size is ignored.

FLEX-ES defaults to *writeback* cache operation. This allows the S/390 disk write channel operation to complete when the data is in the FLEX-ES cache. The data will be flushed to the Server disk at an indeterminate time in the future.[10] An exposure exists if the system fails after a S/390 channel program thinks it completed a disk write, but the cache buffer has not really been written to disk. In this case, the S/390 program may have wrong state information. Such failures are extremely rare and the performance advantage of *writeback* is so great that this default operation is almost always used.

A *writethrough* operation means that the S/390 channel operation for a disk write is not complete until the data is actually written to the Server disk.[11] A copy of the data is retained in the FLEX-ES cache for possible future use. A *writethrough* cache provides considerably lower performance than a *writeback* cache, but it provides higher integrity. It might be considered for a S/390 volume containing DB2 log data, for example.

If you have enough Server memory, you can specify large disk caches for better overall system performance. There is obviously room for considerable tuning here, by manipulating cache sizes at the device and control unit level. You can use the **d ckdcachestats cuu** command to monitor cache effectiveness:

```
flexes> d ckdcachestats A80
ADDRESS  READS  WRITES   CACHE HITS  DEDICATED LINES  LINES USED
  A80    2880   182      1811 (97%)          15         15 (0%)
  A81    2880   182      1811 (97%)          15         15 (0%)
  A82    2880   182      1811 (97%)          15         15 (0%)
```

[10] This should be a familiar concept. The typical UNIX operation involves writeback caches, where the disk cache buffers are synched (flushed) to disk every 10 seconds or so.

[11] This is not quite correct, because the RAID adapter also has a cache and the individual disk drives often have a buffer that performs a temporary cache function. We ignore these points in the current discussion.

The command can specify any device address on the control unit. The *dedicated lines* column indicates the number of dedicated (non-floating) cache tracks. A **clear ckdcachestats cuu** command can be used to reset the statistics.

You can display the definition of an emulated disk with the **d devstate cuu** command:

```
flexes> d devstate A87
Filename: /s390/OS39HA State: OPEN, READY
Options: trackcachesize=30
```

## 5.11 Tuning cachesize

The FLEX-ES *cachesize* parameter (in the system definition section) specifies the number of S/390 bytes that should be reflected in the FLEX-ES instruction cache. FLEX-ES, in effect, compiles S/390 instructions into Pentium instructions in order to execute them. This involves overhead that can be reduced by saving the compiled instructions. This is the purpose of the instruction cache. The underlying structure is complex and proprietary to the FLEX-ES product. The instruction cache requires 11 bytes for each S/390 instruction byte cached. A S/390 LA instruction, for example, requires 44 bytes in the cache.

You can monitor the effectiveness of the instruction cache with the **d cachestats** command:

```
flexes> d cachestats
Cache hits (m1): 1234567/6543321  55%
Cache hits (fb): 66554433/3344556678 20%
Cache hits (fbt): 98744/4567890 33%
Cache misses: 5023567432/103543345876 5%        <== monitor this number
Cache compiles: 2345245/34567346 80%
....
```

The number of *cache misses* in this report is the critical information. If this number is above about 4%, you should increase your *cachesize* parameter. (But never increase it to the point where Linux starts paging!) If the number is considerably less than 4%, you might make better use of your Server memory by increasing the defined S/390 memory size, defining more S/390 expanded memory, or increasing disk cache sizes. The CLI command **clear cachestats** will reset the statistics.

FSI has not documented the meanings of the other statistics in this report.

## 5.12 FSI Channel Adapters

The following FSI hardware adapters are not supported for ThinkPad/EFS systems:

- Parallel Channel Adapter (both versions)
- ESCON Channel Adapter
- ICA Adapter

The required FLEX-ES packages (modules) to use these adapters are not provided with ThinkPad/EFS systems. There are a number of reasons for this, including the following:

- ThinkPad/EFS systems are not intended for production use. Different users will define *production* in different ways, and we will not attempt to make a rigorous definition here. Supporting these adapters could place ThinkPad/EFS in a potential production environment, and this has too many exposures (especially in the RAS area).
- These adapters would require docking stations that could accept full-length PCI adapters. None of the current docking stations *intended for the supported ThinkPad models* have this capability.

- Extensive testing would be necessary to verify PCI bus functions with the FSI adapter cards in this environment. This has not been done.

## 5.13 SCSI adapter for the ThinkPad

Potentially, one might provide an SCSI interface to the ThinkPad and connect external SCSI tape drives. The supported ThinkPad models do not have this capability, but there are third-party PCMCIA cards that might be used. We did not attempt to use a SCSI PCMCIA adapter during our projects. If you require this function, we suggest you contact your business partner or FSI.

## 5.14 Disk fragmentation

The *raw disks* used with UnixWare-based EFS to hold emulated S/390 volumes have a unique advantage. A raw disk is contiguous space on a hard disk drive. Linux file systems do not necessarily provide contiguous space for Linux files. After a Linux file system has been used for normal file creation, deletion, and general operations, it would be unusual for a large file (such as an emulated S/390 volume) to be created in contiguous space in the file system.

FLEX-ES performance benefits from contiguous disk space for an emulated volume. FLEX-ES typically performs Linux disk operations in units of a full track for the emulated volume. For an emulated 3390, this would be 57 KB. If the data is fragmented in different areas of the Linux file system, the I/O operation takes longer.

The standard Red Hat 7.1 distribution does not contain a standard **`defrag`** command. Several defrag-type utilities are available from various Web sources; we did not have the time or resources to thoroughly test any of these and did not use them.

We took the following approach:

- A file system performs contiguous file creation when the file system is new.
- We placed S/390 volumes in separate file systems (/s390 and /s391).
- Generally, we did not place anything else in these file systems. (There are always exceptions, of course, and several of our backup/restore methods used files in these file systems.)
- The slight fragmentation caused by, for example, deleting three (discontiguous) 3390-1 volumes and creating a 3390-3 volume is not significant.

More work is needed to understand the effects of fragmentation on FLEX-ES operation and to document techniques to avoid problems in this area.

## 5.15 Backup and restore considerations

Since a typical[12] ThinkPad/EFS cannot directly connect to tape drives, the backup options are more limited than for a Netfinity/EFS system. Still, backing up and restoring S/390 data has interesting variations. The most basic element involved is where to store your backup data. There are several options:

[12] A ThinkPad/EFS system might use *remote FLEX-ES* resources to connect to a machine that does have attached tape drives. This should work (although we did not try it), but cannot be considered a typical ThinkPad/EFS environment.

- You can store S/390 backups on the ThinkPad disks, as Linux files. This provides logical backup of S/390 data, but does not provide protection against physical failure of disk drives.
- You can **ftp** back up data (in a variety of formats) to another system. This provides protection against physical system failures.

There are two general formats for backed up data:

- A file at the Linux level

  These files contain a complete emulated S/390 volume. It is easy to restore the complete volume, but quite messy to restore a single S/390 data set.

- A S/390 backup, using the OS/390 program ADRDSSU or equivalent

  This backup would be written to a FakeTape file. Once written, this file can be treated as a normal Linux file. It could, for example, be **ftp**ed elsewhere for safety. This type of backup requires the use of OS/390 jobs and may be slower than purely Linux backup functions. It has the strong advantage that single S/390 data sets can be restored easily.

## 5.15.1 Using tar to back up S/390 volumes

An emulated S/390 volume, such as a 3390 volume, is typically a single Linux file. You can back up this volume simply by copying the Linux file somewhere else in your Linux file system. It would be advisable to copy it to another physical disk drive, if possible, for obvious reasons. Emulated volumes are large Linux files and a ThinkPad has limited disk space. Using a compression program as part of the backup process is attractive.

We performed a small test, as follows:

```
$ tar -cvzf /holding/OS39RA.tarz /s390/OS39RA                note 1
tar: Removing leading '/' from member names
$ ls -al /holding                                            note 2
$ cd /s390
$ mkdir untar                                                note 3
$ cd untar
$ tar -xzkwvf /holding/OS39RA.tarz                           note 4
extract 's390/OS39RA'?  y
$ ls -al
$ cmp -s /s390/OS39RA /s390/untar/OS39RA && echo 'OK'        note 5
cmp /S390/OS39RA: File too large
```

1. We used a **tar** command to archive the file into a compressed tar file, which we named *OS39RA.tarz*. The z option in the **tar** command requests compression. The tar/compression step took about 12.6 minutes. The file was saved as a relative name (without the leading /).
2. The **ls -al** command showed the resulting tar file was 468,091,705 bytes. This represents a 6.25:1 compression. Based on PC experience when building AD CD-ROM systems, we typically have approximately a 5:1 compression ratio when compressing files with the zip product for PCs. We have seen ratios from 3:1 to as high as 10:1.
3. We created a new directory, *untar*, so we could recover the file without destroying the original.
4. We changed to */s390/untar* and issued another **tar** command to extract the file. The additional flags (k w) provided reassurance that we would not overlay our original file. The tar extraction placed the extracted file in the current directory (/s390/untar), but with the additional s390 directory name, and an **ls -al** command confirmed it was the correct size. The tar extraction took about 4.75 minutes.

5. We intended to use the **cmp** command to verify that the restored file matched the original file, but it appears that this command does not work with files larger than 2 GB.

The complete operation took place on a single HDD in the ThinkPad. Linux disk buffering was very effective and HDD seeking was not excessive during these operations, judging both from the sound of the HDD and observed performance.

The disadvantage of this backup method is that single S/390 data sets cannot be restored from the backup file. (Of course, you *could* restore the file containing the S/390 volume, rename the file, add it to your FLEX-ES definitions, IPL, clip the S/390 volume to a different volser, vary it online, and copy a single file from it. This is an extended process, but it does work for simple data sets. Attempting to deal with VSAM data sets, for example, could be more complex.)

We did not try this, but we could have first backed up the S/390 volume (with ADRDSSU) to create a FakeTape file and then processed this file with tar. This would require more processing and more Linux disk space, but this process could be used to restore individual OS/390 data sets.

## 5.15.2 Using tar and ftp

One way to back up S/390 volumes on a ThinkPad/EFS system is to **ftp** the Linux file containing the volume to another system. Using a 100 Mpbs LAN connection, especially on a private or local LAN, performance can be quite good. These files are large, by any standard, and ftp'ing compressed files probably makes more sense. We created the following shell scripts:

```
Shell script /usr/flexes/rundir/buOS39RAc

   tar -cvzf /holding/OS39RA.tarz /s390/OS39RA
   ftp -niv 0<buOS39RAp

Shell script /usr/flexes/rundir/buOS39RAp

   open 9.12.17.220
   user ogden xxxxxx
   bin
   put /holding/OS39RA.tarz /s390/TP/OS39RA.tarz
```

We then executed these functions:

```
$ cd /usr/flexes/rundir
$ sh buOS39RAc
tar: Removing ‘/’ from member names
s390/OS39RA
Connected to 9.12.17.220
220 nfefs1 FTP server (Version wu-2.6.1-16) ready
530 Please login with USER and PASS.
530 Please login with USER and PASS.
KERBEROS_V4 rejected as an authentication type
331 Password required for ogden.
230 User ogden logged in/
200 Type set to I.
local: /holding/OS39RA.tarz  remote: /s390/TP/OS39RA.tarz
226 Transfer complete
468091178 bytes sent in 57 seconds (8e+03 Kbytes/s)
221-You have transferred 468091178 bytes in 1 files.
221-Total traffic for this session was 468091705 bytes in 1 transfers.
221 Thank you for using the FTP server on nfefs1
```

The tar function again took 12.6 minutes and the ftp function took less than one minute. We used a private 100 Mpbs LAN connection, with no other active systems on it. The target system (our Netfinity/EFS machine) was otherwise idle during this time. It would have been faster to **ftp** /s390/OS39RA than to **tar** it, but we wanted the disk compression offered by **tar**. We created the /s390/TP directory on the target system before we ran the script.

We elected to use a separate tar file for each volume. A single, combined tar file would be very large and too unwieldy. We simply made more shell scripts; two for each S/390 volume we wanted to back up.

We could, of course, take ADRDSSU backups (to FakeTape files) and then **tar** and **ftp** these files. This might be the best solution, although it would require the most processing and ThinkPad disk space.

There are clearly several ways to approach backup and restore for a ThinkPad/EFS system. We cannot recommend any as the *right* way, and you will probably use a combination of techniques. We strongly suggest that you take time to understand the options available, try whatever is appropriate for you, and establish a routine for taking backups.

This discussion has not mentioned backing up the Server operating system. Ideally, we should be able to capture the complete Server state in a form that can be restored by a standalone boot program. The options are somewhat limited, since our ThinkPad/EFS systems do not have tape drives or CD-RW drives. One option would be to completely duplicate the primary ThinkPad hard disk contents onto another hard disk in the UltraBay. We did not have time to explore this problem during this redbook project, but hope to address it in a future project.

### 5.15.3 CD-RW drive

At least one of the base ThinkPads used for EFS systems has an optional CD-RW drive. This obviously can be used for backup purposes, working from the Linux side and taking full-volume backups. It would be necessary to compress the S/390 volume first, to fit on a CD. Also, some disk staging may be necessary if two hard disks are used for EFS operation and one of these must be removed to mount the CD-RW drive in the Ultrabay.

## 5.16 S/390 identification

The S/390 instruction STIDP stores an 8-byte field:

```
0APLSSSSIIIIOOOO
 |||  |   |   +---> the four zeros are constants
 |||  |   +-------> machine type  (see notes below)
 |||  +-----------> arbitrary number set by manufacturer (serial number)
 ||+--------------> LPAR number
 |+---------------> processor number
 +----------------> the hex digits 0A are normally constants
```

The processor type field for an EFS system has one of these values:

```
1245 is the processor type for a system based on xSeries 430
1247 is the processor type for a system obtained through PID
1246 is for all other EFS systems
```

The SSSS field is normally set to a unique value for each license and functions as a CPU serial number. The LPAR number is usually zero, but you can set it by a statement in the system definitions. The processor number corresponds to one of the processor numbers in your system definitions. For a single processor, this is normally zero.

In most cases, these fields are set by the FLEX-ES license key and you cannot change them. There is an option to make these fields changeable; this option must be specified when the system is ordered. If this option is used, the machine type cannot be set to 1245, 1246, or 1247. Two CLI instructions are associated with this area:

```
flexes> display cpuid
flexes> set cpuid            (only if permitted by the license key)
```

## 5.17 Display PSW and registers

You may sometimes need to display the S/390 PSW. You can do this from the *flexes* prompt provided by the CLI program:

```
flexes> d psw            (displays psw and next instruction, if possible)
flexes> d g              (displays general purpose registers)
```

## 5.18 RAS discussion

Is a ThinkPad/EFS system as reliable as a "real" S/390? *No.* A reliability discussion can quickly become extended and open ended. We need to include Netfinity/EFS systems as part of the following discussion. We can summarize our views this way:

- A Netfinity/EFS or a ThinkPad/EFS system is not a S/390 (or a z900) system in terms of reliability. It does not have dual instruction units in each processor, or an Application Preservation function, for example.
- A Netfinity/EFS system is based on a very high-quality Server and is expected to exhibit much better reliability than a typical desktop PC. Contributing factors are:
  - ECC memory

    This is extremely important in preventing the random lockups and crashes that many PCs experience.
  - Use of a RAID adapter
  - ECC memory in the RAID adapter, for the same reasons
  - Battery operated cache in the RAID adapter
  - Multiple, redundant power supplies
  - Selection of a *very* stable Server operating system (UnixWare)
  - Absence of random applications (of uncertain trustworthiness) on the Server

    (That is, games are not downloaded from random sources on the Web, we hope.)
  - Thorough physical inspection of the Netfinity hardware by the business partner involved

    While this sounds trivial, it contributes to stability.
  - Using a UPS system "in front" of the Server
  - Better cooling

    This is a significant factor for a system that is intended to run continuously.
- A ThinkPad/EFS is based on a quality laptop system, but this does not compete with a Netfinity/EFS server in many RAS characteristics.
  - The ThinkPad does not have RAID protection for its disk(s). With a single disk drive, or, at the most, two drives, there is no practical method to add RAID protection.
  - Multiple power supplies are not available.
  - ECC memory is not used, for a number of reasons.

- Interfaces (such as SCSI or S/390 channels) are not typically available to ThinkPad/EFS users, making routine system backup functions more difficult.
- Cooling effectiveness, especially when used without a docking station, can be greatly influenced by the exact location and positioning of the ThinkPad.

If you *must* have S/390 RAS, you need to buy a S/390. We believe a Netfinity/EFS system will provide more than adequate reliability within its place in the hierarchy of servers. A ThinkPad/EFS system is not intended as a production system and should not be compared with the RAS characteristics of production systems.

## 5.19 Verify ckd disk

The **chkckd** command can be used to verify the internal format of an emulated S/390 ckd disk:

```
# ckdchk -a /s391/WORKO1              (Use your correct file name, of course)
```

The -a flag indicates that a full range of checks should be performed. Any error messages from this utility should be taken seriously. However, the following message might be expected from volumes containing VM minidisks:

```
$ ckdchk -a /s390/OS39M1
FSIDU166 [cyl = 99 head = 14 rec = 1] Record cylinder number on "OS39M1"
does not match home address cylinder number (hacyl: 0x0063 reccyl: 0x0d70)
```

We also get these messages for some of the OS/390 AD volumes built for P/390 emulated I/O; we do not know the cause, but have not had any other problems with the volumes.

If an emulated ckd drive contains improper formatting, *something* is very likely to go wrong during S/390 operation. This can range from crashing one of the FLEX-ES components to strange messages from OS/390 IOS.

The only recovery possible is to restore the volume from a backup of some type.

## 5.20 Linux windows

We found that we often had four windows (terminal emulation progrms) open on the Linux desktop:

- A window running **flexescli** (with the *flexes* prompt)
- A Linux command line window with **su** to *root*
- An x3270 session for the OS/390 master console
- An x3270 session for TSO

The *root* window was useful for issuing **resadm** commands. The window with the *flexes* prompt is generally required for FLEX-ES operation. We found that having two 3270 sessions was our most useful arrangement.

Our typical startup process went like this:

```
Log into Linux with userid flexes         (working directory is /usr/flexes)
Start two terminal windows on the Linux desktop
Window 1
   $ su                                   (enter root password)
   # cd rundir                            (change to /usr/flexes/rundir)
   # resadm -s R1OA.rescf                 (start FLEX-ES resources)
   # resadm -r                            (verify resources started)
Window 2
```

```
   $ cd rundir                          (change to /usr/flexes/rundir)
   $ sh shos                            (start our shell script)
Wait for 3270 sessions to start
   flexes> ipl a80 0a8200               (note that prompt changes to flexes)

Operate OS/390 as normal, using 3270 sessions.  Shut down OS/390 when finished.

Window 2
   flexes> shutdown                     (stop S/390 emulation)
Window 1
   # resadm -T                          (stop FLEX-ES resources cleanly)
   # resadm -k                          (terminate resadm)
   # exit
   $ exit
```

This process can be automated or abbreviated in a number of different ways. We wanted to keep a straightforward, manual control over FLEX-ES startup and used the steps shown here.

Another window (not *root*) was sometimes useful for running **vmstat** commands and similar utility functions. (This could be done from the *root* window if you ignore the usual advice to avoid *root* usage except when specifically required. Or, you could **exit** from root in Window 1 and work from there.)

## 5.21 Installing FLEX-ES upgrades

Installing a new FLEX-ES release under Linux involves a few **rpm** commands:

```
# rpm -e ftlib
# rpm -e msgmgr
# rpm -e flexes
# mount /dev/cdrom /mnt/cdrom            (source for new FLEXES)
# cd /mnt/cdrom
# rpm -i flexes-6.0-5.i386.rpm
(a few messages appear)
# rpm -i msgmgr-6.0-5.i386.rpm
(a few messages appear)
# rpm -i ftlib-6.0-5.i386.rpm
```

The old FLEX-ES packages should be removed (with the -e option) in the order shown, and the new packages installed in the order shown. Your package names may be different (to reflect new release numbers) and your source might be ftp files rather than a CD-ROM, so you need to adjust the exact commands shown above to match your environment. When removing the old packages, you can ignore a message about not removing the /usr/flexes directory.

## 5.22 ThinkPad power control

After installing Linux on our ThinkPad, we found that the *suspend* function (that is, closing the “lid” of the ThinkPad) worked correctly when Linux and FLEX-ES were running. However, if we left the system alone while running, with no keyboard or mouse movement for several hours, it automatically removed power and we needed to reboot Linux. We were unable to find any power controls in Linux that managed this timeout function.

We went into the ThinkPad BIOS (F1 after starting the ThinkPad) and changed all the power-related functions to “Disable.” This solved the problem and we could leave OS/390 running overnight with no problems. We did not attempt to analyze the exact use of each BIOS control. We needed to scroll in at least one of the BIOS screens to access all the controls.[13]

## 5.23 Useful Linux commands

Not all ThinkPad/EFS owners have Linux or UNIX backgrounds, and a few brief notes may help these users.

### Common commands

New users may find Linux commands a little confusing. The following is intended as a brief reminder of the syntax for a number of commands often used while working with Linux in a ThinkPad/EFS environment.

```
chown flexes /s390/volA          make userid “flexes” owner of this file
chgrp flexes /s390/volA          make group “flexes” the group owner of this file
chown aaa:bbb /home/myfile       make sss the owner userid and bbb the owner groupid
chmod 600 /s390/volA             allow only file owner to read & write this file

cat filename                     list all of a file on the terminal
more filename                    list a file, page by page (space bar to advance page)
                                 (Ctrl-z to exit from the program)
less filename                    a newer version of more

cp /s390/volA /s391/volX         copy first file to second file
find / -name unzip               start search in root and find file named “unzip”

updatedb                         update the database used by the locate command (root)
locate unzip                     locate all files with “unzip” in their name
file /usr/bin/unzip              describe nature of named file

mkdir /home/ogden/source         create a new directory
ls -al                           list current directory, in some detail
ls -lrt                          list current directory, most recently updated files last
mv oldfile newfile               rename or move a file
rm filename                      delete a file
head filename                    list first 10 lines of file
tail filename                    list last 10 lines of file

tar -cvxf mytar.tar /flexes/rundir       create tar file from files in /flexes/rundir
tar -tvxf mytar.tar                      list files contained in “my.tar”
tar -xvxf mytar.tar                      extract all the files in the tar file

unzip /dev/cdrom/os390/os39ra.zip  -d/s390         unzip a file on CD-ROM

mdir a:                          display DOS diskette
mcopy /flexes/x/y a:x.txt        copy Linux file to a DOS diskette file
mdel a:afile.txt                 delete a file on diskette
mformat a:                       format a DOS diskette

shutdown -h now                  shut down the Linux system
```

[13] When we first installed the ThinkPad, we quickly checked the BIOS controls and thought we disabled any automatic power-off function. However, we failed to scroll through all the options and we had the automatic shutdown problem just mentioned.

```
last                              who has logged into Linux?
w                                 who is logged into Linux now?
whoami                            displays your userid

ifconfig                          display current LAN adapter status
netstat -r                        display current routing state
route add default gw 9.12.17.150 eth0          create default gateway
route add -net 10.10.0.0 255.255.255.0 eth0    add network route

top                               monitor activity.  "q" to quit
vmstat 10 5                       5 activity reports.  10 seconds between reports.

mount /dev/cdrom /mnt/cdrom       mount a CD-ROM
mount /dev/fd0 /mnt/floppy        mount a diskette (but not a DOS diskette)
mount /dev/hdc2 /s391             mount file system on second HDD at mount point /s391
mount                             display the mount table
```

## Multiple consoles, sessions, screens

Linux offers multiple consoles on the PC (or ThinkPad) display. The keys Ctrl-Alt-Fn are used to select a console. Ctrl-Alt-F1 through Ctrl-Alt-F6 produce traditional UNIX-style command lines, and Ctrl-Alt-F7 produces the X windows GUI desktop. Each of these consoles is a separate Linux *session*, requiring a separate login. You can log in with different userids in each session, or use the same userid in several sessions. You can switch between sessions at any time, using Cnrt-Alt-Fn keys.

You can also have multiple command line windows open on the desktop. These do not require separate logins.

## Text editors

Traditional UNIX users and many Linux users regard **vi** as their primary text editor. Traditional mainframe users often avoid **vi**, if possible. The kde desktop offers the *Advanced Editor*, which can be started from the *fountain pen nib* icon in the toolbar. This is much like a simple PC text editor and very unlike **vi**. It is suitable for almost all the text editing we required while installing and customizing FLEX-ES. Anyone comfortable using any version of Windows or OS/2 should have no trouble using this editor. The Red Hat Linux distribution includes a number of editors, and you can select the ones you like best.

6

# Frequently asked questions

**Q:** Can I install a boot manager and have Linux (with FLEX-ES) and Windows on the same ThinkPad**?**
**A:** Yes, but you will need to do the installation yourself. You might consider completely removing the primary hard disk and replacing it with a disk containing whatever version of Windows you prefer. However, the primary hard disk (as opposed to a second hard disk) is not intended for frequent removal.

**Q:** Does each S/390 user need a Linux userid?
**A:** No. A typical OS/390 TSO user, for example, never “sees” Linux.

**Q:** Must I use **`vi`** to administer the Linux system?
**A:** No. Linux has a number of other editors, some of which are very “PC-like.” However, you may find it easier to use **`vi`** to edit files whose names begin with a period.

**Q:** Why do I need to define emulated control units and channels? I did not need to do this with emulated I/O on P/390 and MP3000 systems. For these machines, I could simply define emulated devices.
**A:** There are two major reasons. The first is that working through (emulated) channels and devices provides a more accurate emulation of S/390 hardware. In most cases this has little practical effect. The second reason involves emulating multiple S/390 instances, and sharing emulated I/O--for example, shared DASD. This is a more complex situation that does not exist on a P/390 and is not permitted (via LPARs) on an MP3000. The emulated control units provide the logical interlock and queueing points for sharing devices according to S/390 architectural specifications.

**Q**: How real is the channel emulation? Can I make my system faster by defining lots of channels and *spreading out* my I/O devices?
**A:** As best we could tell, the emulated channels are not very real in this sense. Many system definitions we examined define only two or three channels of each channel type. We have noticed that it appears necessary to define separate channels for emulated 3174 or 3274 control units.

© Copyright IBM Corp. 2001

**Q:** I understand that a LAN MAC address is used to control FLEX-ES licenses for Netfinity Servers. Can I use the same method for ThinkPads?
**A:** No. FLEX-ES licenses for ThinkPad/EFS require the use of a *dongle*. In general, any FLEX-ES version that uses Linux as the base operating system will require a dongle.

**Q:** Can I run a ThinkPad in 24x7 operation? (24 hours/day, 7 days/week)
**A:** There is no rule against it, and some ThinkPads are left running in docking stations. However, *in our opinion*, this is not a reasonable solution for an EFS system. The fully-loaded ThinkPad used for EFS develops considerable heat (on the bottom) and may require additional air flow. In addition, a system intended for 24x7 S/390 operation should have RAID disks.

**Q:** Will the FLEX-ES code for a Netfinity/EFS system run on a ThinkPad?
**A:** There are really two questions here. The UnixWare version of FLEX-ES will not run on Linux. (Netfinity/EFS is currently based on UnixWare and ThinkPad/EFS is based on Linux.) Can UnixWare be used on a ThinkPad? We assume it can, but have not tried it. If so, then the Netfinity/EFS FLEX-ES code could be used, although you would require a new FLEX-ES license that matches a MAC address in the Thinkpad *and* that matches the MHz speed of the ThinkPad. This combination is not supported at this time.

**Q:** Can I use token ring LAN adapters? You mention only Ethernet.
**A:** Yes, token ring can be used. We usually discuss Ethernet because (1) it is what we used, and (2) the majority of EFS users will probably have Ethernet LANs. You should obtain the current status from your business partner.

**Q:** How many Ethernet adapters should I have for the ThinkPad?
**A:** One should be sufficient, and will probably be the mini-PCI adapter for the ThinkPad. You can share this adapter (using multiple IP addresses) between Linux and S/390 operating systems.

**Q:** Can I use the FSI channel adapters or ICA adapter with a ThinkPad? I could put them in a docking station.
**A:** No. This is not supported.

**Q:** Can I have “odd size” 3390 drives, such as a 100 cylinder 3390 on a P/390?
**A:** No. Only standard size drives are emulated.

**Q:** An emulated 3390-3 was 3339 cylinders on my P/390. Why is it 3343 cylinders under FLEX-ES?
**A:** A standard 3390-3 has 3339 cylinders of directly usable space. The additional cylinders include the “CE cylinder” (for testing) and alternate tracks. P/390 emulation did not include these additional cylinders, while FLEX-ES emulation does include them. In normal use, you will not see any difference between the two methods.

**Q:** Are 4mm tapes compatible between my MP3000 and my Netfinity/EFS system?
**A:** Yes, they should be, for those EFS systems that support 4mm drives. (ThinkPad/EFS does not support SCSI tape drives.)

**Q:** How many emulated S/390 devices can I have?
**A:** There is a FLEX-ES maximum of 2048 emulated devices per S/390 instance. If you emulated two S/390s, each will have this maximum number of devices.

**Q:** I have a fixed amount of real memory in my ThinkPad. You say that I should not overcommit this for S/390 emulation. Should I make my emulated S/390 central storage as large as possible (without triggering Linux swapping) or should I define some expanded storage at the expense of central storage?
**A:** Provided you do not exceed the maximum 2 GB permitted for central storage, we think

that Server memory is better used as S/390 central storage than as S/390 expanded storage. However, you might have specific applications that require expanded storage, so we cannot provide an absolute answer. You face the same quandary when dividing up real memory on an MP3000, for example, between central storage, expanded storage, and a disk cache.

**Q:** Several ThinkPad processor speeds are available. The model you used for this redbook (750 Mhz) is one of the slowest. Can I use a faster processor?
**A:** Yes.

**Q:** Can I write FakeTape files from my UNIX and Linux applications?
**A:** Yes. The format is well documented. APIs are provided if you work under UnixWare and Linux.

**Q:** How many x3270 sessions or TN3270 sessions can I have on the ThinkPad display?
**A:** There is no particular limit. Reasonable screen usage might indicate a practical maximum of something like three or four.

**Q:** Should I use Terminal Solicitor sessions (via Linux TCP/IP) or OS/390 TCP/IP connections for my TSO and CICS users?
**A:** Assuming OS/390 TCP/IP is configured and working, it does not appear to make much difference. OS/390 TCP/IP connects TN3270 sessions to VTAM and whether VTAM is using a local, non-SNA 3270 (as emulated by the Terminal Solicitor) or working through OS/390 TCP/IP has no observable effect at the end-user level. Systems programmers will probably prefer to work through the Terminal Solicitor because it is operational even if OS/390 TCP/IP is not working.

**Q:** When I connect via OS/390 TCP/IP, I do not get the terminal logo I always receive if I connect via the Terminal Solicitor. Why not?
**A:** This is due to your OS/390 TCP/IP setup. The OS/390 TCP/IP PROFILE needs to point to a USSTAB module that generates the logo. Recent AD releases have this set up.

**Q:** Can I connect to OS/390 with an ASCII telnet session? I want to use `vi`.
**A:** Yes, although this is not really a ThinkPad/EFS question. Recent AD releases have port 1023 (of OS/390 TCP/IP) configured for ASCII telnet sessions. If you are not using an AD release, you will need to configure OS/390 TCP/IP.

**Q:** I have the impression there are frequent FLEX-ES releases. Do I receive these automatically? When should I upgrade? Where is there a concise list of what is new in each release?
**A:** In general, releases are obtained through the IBM business partner from which you purchased the system. You should upgrade only when a later release contains a feature you need. Information on what is new in each release can be obtained from your IBM business partner.

**Q:** I have the latest z/OS AD release. Can I run it on a ThinkPad/EFS?
**A:** Yes. The latest RELS-ES release (at the time of writing) supports z/OS in 31-bit mode.

**Q:** Where can I order a printed copy of the FLEX-ES manuals?
**A:** FLEX-ES manuals can be obtained from the IBM business partner from which you purchased the system.

**Q:** Can I use the ThinkPad for other purposes when FLEX-ES is not running? Can I run Linux server functions on it when FLEX-ES is running?
**A:** When FLEX-ES is not running, you can run anything you want on your Thinkpad—taking care not to disturb the FLEX-ES customization. When FLEX-ES is running, you could still run other Linux workload. However, we generally recommend against this. The external workload

may force Linux swapping and this has severe effects on FLEX-ES performance. Having gone to considerable expense to obtain a small S/390 (in the form of your ThinkPad/EFS), we suggest you use it only as a S/390. If you need to also run a Linux server, we suggest you obtain another machine.

**Q:** Should I purchase an Uninterruptable Power Supply (UPS) for use with my ThinkPad/EFS machine?
**A:** The ThinkPad battery provides, in effect, a UPS function. You might consider an external surge protector.

**Q:** How do I order a ThinkPad/EFS system? What if I am not a PID member?
**A:** You do not need to be a PID member to order a ThinkPad/EFS system (although you will not have access to AD CD-ROM systems). The FSI Web site has a list of their business partners.

**Q:** Where can I find documentation about the various FLEX-ES traces?
**A:** There are several FLEX-ES traces:

- The *processor event trace* is always active and uses a permanently defined circular buffer. It is intended only for FLEX-ES debugging and is not documented.
- The *processor instruction trace* uses the tracesize buffer you define in your system definition. It is intended for FLEX-ES debugging but might be useful to more sophisticated customers. It is minimally documented at present, but FSI intends to provide better documentation.
- *Device traces*, which are usually obtained with the **`devtrsnap`** command. These are not documented at present, but a user familiar with channel programming techniques might find these traces useful.

The traces are not well documented at this time.

**Q:** Are any modifications, PTFs, or SPEs required for OS/390 on EFS?
**A:** No.

**Q:** Are printer FCBs emulated?
**A:** Yes. FLEX-ES printer emulation supports the Load Forms Control Buffer channel command. The default FCB provides 66 lines at 6 lines/inch, with a “channel 1 punch” at line 3 and a “channel 12 punch” at line 64.

**Q:** Is a default 1403 carriage tape emulated?
**A:** Yes. It is set for 66 lines at 6 lines/inch, and has a channel 1 “punch” at line 3. There is no way to alter these settings.

**Q:** Are Token Ring and Ethernet equally supported?
**A:** No. (We were still researching this question at the time of writing.)

**Q:** Can I manage an ESCON director from an EFS system?
**A:** Not from a ThinkPad/EFS system!

**Q:** I am interested in using FLEX-ES remote resources for disks and tapes. Do I need a FLEX-ES license just to run the resource manager on remote systems?
**A:** Yes, you will need a FLEX-ES license for every system running the resource manager.

**Q:** If I obtain faster ThinkPads, can I ask FSI to alter my license? Do I need to notify IBM?
**A:** Yes, you can ask FSI for a new license key. (You *must* do this because your old key will not work with the faster processors.) You do not need to purchase a new license, assuming you have retained the same dongle. FSI does not notify IBM or any other software vendor. You need to manage any potential S/390 software fee changes related to the increased MIPS of your new machine. This is obviously a complex area, and these processes may change in the future.

**Q:** What is the difference between swapping and paging?
**A:** None, in these discussions.

**Q:** You mention MIPS in the 10-12 range; business partners talk about 16-20. Which is correct?
**A:** Both and neither. An emulated S/390 has a much wider range of MIPS than a “real” hardware S/390. Any MIPS measurement is very strongly dependent on the instruction mix and the addressing mix involved, as well as the emulation techniques used. We could probably contrive code to produce anything from below one MIPS to 30+ MIPS. We try to use conservative numbers in this series of redbooks, while marketing representatives tend to use optimistic numbers. An exact MIPS number simply does not exist.

**Q:** How is floating point performance?
**A:** It is acceptable for casual use, but you would probably not select this platform for major floating point applications. Binary (IEEE) floating point has considerably better performance than hex floating point.

**Q:** There are always new Linux functions available, as well as kernel fixes and upgrades. Should we install these? Do we need to reinstall any part of FLEX-ES afterward?
**A:** If you have a stable system, we suggest you do not apply Linux changes. You should regard your EFS system as a S/390 and not as a Linux machine. If you do rebuild the Linux kernel, you will need to reinstall the FLEX-ES msgmgr package.

**Q:** How can I tell whether my Ethernet connection is running at 10 or 100 Mbps?
**A:** We did not find a convenient way to do this at the ThinkPad end. Many Ethernet hubs have an LED that indicates 100 Mbps operation and you might look for this.

**Q:** Can I use DHCP to obtain a LAN IP address?
**A:** Yes, you could do this for Linux TCP/IP, but not for OS/390 TCP/IP. However, this is probably not a practical solution. Your EFS system is essentially a S/390 server, not a client, and you will probably need a stable IP address for it.

**Q:** How difficult is it to change the IP address of my system?
**A:** You are likely to have two IP addresses: one for Linux and one for OS/390. To change the OS/390 address you must edit the PROFILE data set and then restart TCPIP. If you are sharing a ThinkPad Ethernet adapter with Linux, you will also need to change the IP address in the FLEX-ES resource stanza that defines the emulated 3172. To change the Linux IP address:

```
(gnome): Programs --> System --> Network Configuration.  Select the Interfaces tab.
         Deactivate the appropriate interface.  Edit the interface to change the IP
         address.  Activate the interface.  Save.  Quit.
```

**Q:** As a developer, I sometimes need a real hard-copy listing. You provide very little information about setting up printers. Why?
**A:** We did not have time to experiment with printers while writing this redbook. This can become a complex topic quickly when network printers are considered. We hope to provide more detail if a follow-on redbook is written. Larger EFS systems often have channel-attached S/390 printers and no special documentation has been required to cover these.

**Q:** How long does the ThinkPad battery last?
**A:** With our fully loaded system (512 MB, two disk drives, heavy computational load) the battery usually lasted about 50 minutes.

**Q:** Is SNA available over Ethernet?
**A:** Not yet, at the time of writing. When it is available, the selection of DIX versus IEEE802.3 frames will be done by selecting an appropriate control unit type in the FLEX-ES definitions. Control unit type *xcasna* will select IEEE802.3, and *xca* will select DIX.

**Q:** Can I use a locally administered MAC address for Ethernet?
**A:** We have never done this, although it appears that some Ethernet drivers appear to support this option.

**Q:** How many Ethernet adapters do I need if I want to use SNA and TCP/IP?
**A:** We do not know yet.

A

# FLEX-ES definition listings

This section lists the FLEX-ES definitions we used when installing our ThinkPad/EFS system and writing this redbook. These files were copied from our working system and represent the system at the last stages of our project.

## Basic definitions for a single HDD

Our system and resources definitions (both in the same file) used for the four volumes required to IPL the OS/390 Release 10 AD CD-ROM system were as follows:

```
system S10A:
memsize(262144)
cachesize(2048)
essize(64)
instset(esa)
cpu(0)
channel(0) local
channel(1) local
channel(2) local
cu devad(0xA80,8) path(2) resource(R10A3990)
cu devad(0x700,3) path(0) resource(R10A3174)
cu devad(0xE20,2) path(1) resource(R10A3172)
cu devad(0x560,1) path(2) resource(R10A3480)
end S10A

resources R10A:
R10A3990: cu 3990
interface local(1)
device(00) 3390-3 /s390/OS39RA
device(01) 3390-3 /s390/OS3RAA
device(02) 3390-3 /s390/OS39M1
device(03) 3390-3 OFFLINE
device(04) 3390-3 OFFLINE
device(05) 3390-3 OFFLINE
device(06) 3390-3 OFFLINE
device(07) 3390-3 /s390/OS39HA
end R10A3990
```

© Copyright IBM Corp. 2001

```
R1OA3174: cu 3174
interface local(1)
device(00) 3278 mstcon
device(01) 3278 M701
device(02) 3278 M702
end R1OA3174

R1OA3172: cu 3172
interface local(1)
options 'ipaddress=9.12.17.211'
device(00) 3172 eth0
device(01) 3172 OFFLINE
end R1OA3172

R1OA3480: cu 3480
interface local(1)
device(00) 3480 OFFLINE
end R1OA3480

end R1OA
```

Some FLEX-ES users elect to define emulated 3270 devices as OFFLINE and then use FLEX-ES **mount** commands (in a shell script) to define and name terminals. For our small system, we found it easier to simply define a number of 3270 terminals (with names such as L701) in our *resources* definition.

The four AD system volumes used for our initial IPL are normally mounted at addresses A80, A81, A82, and A87. The “gap” between A82 and A87 required the definition of several OFFLINE 3390 volumes. (In this case, the device model (3390-1, -2, -3) is not meaningful.) It would be possible, of course, to mount these volumes at any 3390 addresses contained in the OS/390 IODF. We could have placed the A87 volume on A83 and avoided defining the OFFLINE devices. However, other AD CD-ROM documentation (including several redbooks) uses the addresses we selected and we want to preserve compatibility with this existing documentation.

(If the AD CD-ROM system is installed on the logical volumes of a Multiprise 3000 system, it is typically placed at addresses AA0, AA1, and so forth, or sometimes at addresses 300, 301, and so forth. Normal OS/390 rules for device addressing and usage are followed; there is nothing unique about a ThinkPad/EFS system in this respect.)

The tape definition is OFFLINE. This means that FLEX-ES **mount** commands are needed before the device can be used. This is typical, and allows the UNIX files to be associated with such devices to be selected and changed while the OS/390 system is operational.

## Shell script for a single HDD

After using **cfcomp** to compile the above definitions, we used this shell script to start operation. (A **resadm -s R1OA.rescf** command must be issued first.)

```
flexes S1OA.syscf
xmodmap -e 'keysym Alt_L = Alt_L Meta_L'
xset fp+ /usr/flexes/fonts
xset fp rehash
echo 'Verify your master console is active'
x3270 -model 3 -keymap pc -port tn3270 localhost:mstcon &
x3270 -model 3 -keymap pc -port tn3270 localhost:M701 &
flexescli localhost S1OA
```

# Definitions for two HDDs

The following definitions assume that the second ThinkPad hard disk is mounted and contains the optional volumes discussed in text. Since these definitions reference files on the second hard disk (with mount point /s391), they can be used only when the second hard disk is installed (and the file system mounted). Note that the system name is now S10B and the resources name is R10B.

```
system S10B:
memsize(262144)
cachesize(2048)
essize(64)
instset(esa)
cpu(0)
channel(0) local
channel(1) local
channel(2) local
cu devad(0xA80,9) path(2) resource(R10B3990)
cu devad(0xA90,3) path(2) resource(R10B3991)
cu devad(0x700,3) path(0) resource(R10B3174)
cu devad(0xE20,2) path(1) resource(R10B3172)
cu devad(0x560,1) path(2) resource(R10B3480)
end S10B

resources R10B:
R10B3990: cu 3990
interface local(1)
device(00) 3390-3 /s390/OS39RA
device(01) 3390-3 /s390/OS3RAA
device(02) 3390-3 /s390/OS39M1
device(03) 3390-2 /s391/OS39PA
device(04) 3390-2 /s391/OS3PAA
device(05) 3390-3 /s391/OS39DA
device(06) 3390-3 /s391/OS3DAA
device(07) 3390-3 /s390/OS39HA
device(08) 3390-2 /s391/OS3DAB
end R10B3990

R10B3991: cu 3990
interface local(1)
device(00) 3390-1 /s391/WORK01
device(01) 3390-1 /s391/WORK02
device(02) 3390-1 /s391/WORK03
end R10B3991

R10B3174: cu 3174
interface local(1)
device(00) 3278 mstcon
device(01) 3278 M701
device(02) 3278 M702
end R10B3174

R10B3172: cu 3172
interface local(1)
options 'ipaddress=9.12.17.211'
device(00) 3172 eth0
device(01) 3172 OFFLINE
end R10B3172

R10B3480: cu 3480
```

```
interface local(1)
device(00) 3480 OFFLINE
end R10B3480

end R10B
```

## Shell script for two HDDs

This shell script is just like the first one, except that it uses system name S10B instead of S10A.

flexes S10B.syscf

```
xmodmap -e 'keysym Alt_L = Alt_L Meta_L'
xset fp+ /usr/flexes/fonts
xset fp rehash
echo 'Verify your master console is active'
x3270 -model 3 -keymap pc -port tn3270 localhost:mstcon &
x3270 -model 3 -keymap pc -port tn3270 localhost:M701 &
flexescli localhost S10B
```

B

# FLEX-ES parameters

This appendix reviews a selected subset of the FLEX-ES parameters needed to define emulated S/390 systems and their resources. The formal FLEX-ES documentation should be reviewed for additional information, more emulated devices, and many other parameters.

## System definitions

System definitions are usually something like this:

```
system S10A:
memsize(262144)                 # K Central Storage = 256 MB
essize(64)                      # M Expanded Storage = 64 MB
instset(esa)
tracesize(256)
cpu(0) dedicated                # Single Pentium, dedicated
channel(0) local
channel(1) local
channel(2) local
channel(3) local
cu devad(0x00C,3) path(0) resource(R10A2821)    # 2450R, 2540P, 1403
cu devad(0x560,1) path (0) resource(R10A3480)   # tape drive (4mm)
cu devad(0x700,16) path(1) resource(R10A3174)   # 3270 terminals
cu devad(0xA80,12) path(2) resource(R10A3990)   # 3390 disks
cu devad(0xE20,2) path(3) resource(R10A3088)    # CTCs for TCP/IP
end S10A
```

The first line (system S10A) provides a name for the system being defined. The FLEX-ES compiler will create a file named, for this example, *S10A.syscf*.

**memsize** always expresses kilobytes of storage. The largest accepted value is 2097152, which would define a 2 GB system. The size specified will be the size of central storage in the emulated S/390. There is no direct relation to the memory size on the ThinkPad server, although we strongly recommend that `memsize` be such that no paging is required by the underlying Linux system.

© Copyright IBM Corp. 2001

The parameter **essize** specifies expanded storage size and expresses megabytes of storage. Furthermore, the number specified must be an even multiple of 16. If essize is not specified, then no expanded storage is emulated. A specification of 64 means 64 MB of expanded storage.

The **instset** parameter should always be specified as instset(esa) for the OS/390 systems we are discussing.

The **cachesize** parameter specifies the amount of storage to be used by FLEX-ES to cache dynamically compiled S/390 instructions. This parameter expresses kilobytes of *S/390* storage. (The server storage required is 11 times the specified S/390 storage value.) Practical values are in the range 1024 to 8192. For a Thinkpad/EFS system, 2048 appears to be a practical value. We suggest you do not use values outside the recommended range unless you are clearly in an experimental mood. A specification of 2048 means 22 MB of ThinkPad storage will be used by FLEX-ES for a S/390 instruction cache.

The **tracesize** parameter specifies the number of entries in a FLEX-ES trace table. We suggest 512 as a reasonable number for a Thinkpad/EFS system. The FLEX-ES instruction trace function runs only when started. There is no processor overhead caused by defining a tracesize parameter. This is a somewhat imprecise parameter because the size of each entry in the trace table is not well defined. In any event, the total memory used (with the recommended value) is fairly trivial.

There is also a processor *event trace* that uses a permanently defined buffer. This event trace is always active and is not associated with the **tracesize** parameter.

The **cpu** parameters indicate the number of emulated S/390 CPUs and assign identification numbers to each one. Multiple **cpu** parameters are used to specify multiple emulated S/390 CPUs within a single emulated S/390. For example,

```
cpu(0)
cpu(1)
cpu(2)
```

as part of a system definition would specify an emulated S/390 with three CPUs. The CPU numbers (which are defined S/390 architecture elements) are 0, 1, and 2. *The number of CPUs specified may not exceed the number of server processors enabled for S/390 emulation.* For a ThinkPad/EFS system, this will be a single Pentium processor. Therefore, only one **cpu** parameter may be specified. (The single CPU may be given any number in the range 0 - 7, but there is no reason to use anything other than 0.) The **dedicated** keyword on a CPU definition statement means that the underlying ThinkPad processor will be used only for S/390 emulation; it will not be used for Linux work. This option cannot be used with a ThinkPad/EFS system because only one processor is available for both S/390 and other Linux work. Netfinity/EFS systems will normally have at least two Server processors and all but one will be dedicated to S/390 emulation (unless you are running multiple S/390 instances; in that case, the situation is more complex).

The **channel** parameters specify emulated channels and assign channel numbers. For example:

```
channel(3) local
channel(1) local
```

defines two channels, with the channel numbers 3 and 1. The channel numbers must be unique, but need not be in sequence. The **local** keyword means the channels are emulated block multiplexor channels and are emulated using resources on the ThinkPad.[1] The channel numbers (or *path numbers*) are somewhat arbitrary and do not necessarily mirror the channels you might have on a "real" S/390. The channel numbers you assign are *not* reflected in the device addresses ("device numbers") used by OS/390.

The **cu** statement defines an emulated control unit. The **devad** parameter defines the beginning address and the number of (emulated) devices connected to the control unit. This address (or device number) is the address seen by the operating system. For example, if you want to "IPL from A82," then address A82 would appear in a devad parameter.[2] The **path** parameter specifies an emulated channel connected to the control unit. The **resource** parameter specifies the name of a section in a resource definition (described below) that defines the emulated *devices* connected to the control unit. Resource names, used in **cu** statements, must be unique.

The **cu** definition specifies a numberofunits for each cu. You *must* define this number of devices in the matching resource definition. For example:

```
cu devad(0xA80,12) path(2) resource(R10A3990)
```

specifies that 12 devices are attached to this control unit. The R10A3990 clause in the resources definition must contain 12 devices.

An **end** statement is required for the system definition, and the parameter should specify the same name as the *system* statement.

# Resource definitions

Resource definitions refer to actual devices or files and are defined separately from system definitions. In practice, both a system definition and the corresponding resource definition might be in the same text file. Both must be "compiled" by the FLEX-ES **cfcomp** program, to produce *syscf* and *rescf* files, before they can be used.

## Emulated control unit types

FLEX-ES emulates many control unit types (*cutype* parameters) and device types; the full documentation should be consulted for a complete list. The commonly used *cutypes* include:

```
3990            DASD control unit (for all 3380 and 3390 devices)
3480 and 3490   Tape control unit (for FakeTape and SCSI tape drives)
3174            Terminal control unit (for emulated locally-attached 3270s)
3172            LAN control unit (for Ethernet LANs)
3172TR          LAN control unit (for token ring LANs)
3215            Typewriter system console (mostly for VM and Linux for S/390)
2821            Unit record control unit (for emulated card reader/punch, printer)
ctc             Channel-to-channel control unit
```

[1] Other types of channels may be specified, such as byte multiplexor and channels using remote (via TCP/IP) resources. These are described in detail in the FSI documentation.

[2] For this example, we might have **devad(0xA80,8)** as the parameter. This defines 8 addresses, A80, A81, A82, and so forth. A82 is named in this sequence.

## Emulated device types

FLEX-ES emulates many device types, and these are documented in the full FLEX-ES materials. The following list includes commonly used devices for a ThinkPad/EFS running OS/390 or z/OS.

```
3390-1          3390 model 1, with 1117 cylinders (1113 usable cylinders)
3390-2          3390 model 2, with 2230 cylinders (2226 usable cylinders)
3390-3          3390 model 3, with 3343 cylinders (3339 usable cylinders)
3390-9          3390 model 9, with 10038 cylinders (10017 usable cylinders)
3172            LCS device, for Ethernet connections
3172TR          LCD device, for token ring connections
3215            Typewriter console (not for OS/390 or z/OS)
3178            Most typical 3270 definition
3420            "Round" tape, emulated with SCSI device or FakeTape
3480            Emulated with SCSI device or FakeTape
3490            Emulated with SCSI device or FakeTape
3490-E          Emulated with SCSI device or FakeTape
1403            Printer, emulated with UNIX lp functions
2540R           Card reader, emulated with UNIX files
CTC             Channel-to-channel adapter, emulated with UNIX sockets
```

## Typical resource definitions

A simple set of resource definitions, corresponding to the system definitions above, might be:

```
resources R10A
mema: memory
   264
end mema

R10A2821: cu 2821
  interface local(1)
  device(00) 2540R OFFLINE
  device(01) 2540P OFFLINE
  device(02) 1403 OFFLINE
end R10A2821

R10A3480: cu 3480
  interface local(1)
  device(00) 3480 OFFLINE
end R10A3480

R10A3174: cu 3174
  interface local(1)
  device(00) 3278 mstcon
  device(01) 3278 payroll1
  device(02) 3278 develop2
  device(03) 3278 OFFLINE
  .....
  device(15) 3278 sysprogs
end R10A3174

R10A3990: cu 3990
  interface local(1)
  device(00) 3390-1 /s390/.....
  device(01) 3390-3 /s390/.....
  device(02) 3390-3 OFFLINE
  ....
  device(10) 3390-3 /s390/..... devopt 'writethroughcache,trackcachesize=50'
  device(11) 3390-1 /s390/.....
```

```
end R10A3990

R10A3088: cu 3172
  interface local(1)
  options 'ipaddress=9.12.17.211'
  device(00) 3172 eth0
  device(01) 3172 OFFLINE
end R10A3088

end R10A
```

A **resources** block must have a name; R10A is used in this example. Each set of resources in the block must also have a name. Each set name should match a name in a cu statement in the system definition. The general syntax for defining a resource set is:

```
resourcename: cu cutype
   parameters
end resourcename
```

You assign a resource name. It must follow the FLEX-ES rules for names. The name is followed by a colon and the keyword **cu**. The cutype must be one of the cutypes known to FLEX-ES. This is followed by whatever parameters are needed to specify the particular device(s) being defined. The keyword **end** is then followed by the resource name.

The **memory** definition is needed only if your system contains an FSI parallel channel adapter (either model) or ICA adapter. The parameter value is normally 1.03*memsize, where the memsize is specified in the system definition.[3] You will need to consult the full FLEX-ES documentation to understand server memory management when parallel channel adapters are used. If you do not have a parallel channel adapter or ICA adapter, you should omit the memory definition in the resources block.

In the following paragraphs, we show examples of commonly used device resource definitions and describe parameters used for each device type.

## CKD disk resources

```
R10A3990: cu 3990
interface local(1)
device(00) 3390-1 /s390/.....
device(01) 3390-3 /s390/.....
device(02) 3390-3 OFFLINE
....
device(10) 3390-3 /s390/..... devopt 'writethroughcache,trackcachesize=50'
device(11) 3390-1 /s390/.....
end R10A3390
```

The **interface local(1)** parameter means that the defined devices are local (on the same server running FLEX-ES) and that there is one path (channel + control unit) pointing to these devices.[4] Multiple paths are typically found when running multiple S/390 instances (that is, emulating multiple S/390 systems at the same time) and sharing control units among the multiple instances. Shared DASD is the most common example of this.

[3] If multiple S/390 instances are used, the situation is more complex. The maximum memory parameter is the smaller of 512 MB or 3/8 of the server real memory size.

[4] FLEX-ES can use remote resources. For example, a separate UNIX server (connected via TCP/IP) could contain emulated 3390 volumes, but not contain an emulated S/390. We believe that new ThinkPad/EFS users are unlikely to use remote resources while getting started with EFS operation and we do not further describe remote resources in this redbook.

The **device(00)** parameter describes the first device of this resource set. The index number is *decimal*; it must begin with 00 and be incremented by one for each additional device. If your device addressing scheme (at the OS/390 level) requires addressing gaps, you must define dummy (**OFFLINE**) devices in the resource set. The addresses seen at the OS/390 level are set by the *cu* statement (in the *systems definition* section) that points to this resource set.

The device type (**3390-1** and **3390-3** in this example) must be from the list of known FLEX-ES emulated device types. (These are listed in Appendix , “Emulated device types” on page 76.) The next parameter, **/s390/xxxxx**, is the Linux file that contains the emulated 3390 volume. This can be the actual name or a symbolic link.

The keyword **devopt** indicates that optional parameters follow. The **writethroughcache** parameter causes the FLEX-ES 3390 emulation program to consider s S/390 write operation complete when the data is actually written to disk. This is described in “Disk caches” on page 50.

The **trackcachesize=** parameter tells the emulation program to allocate the indicated number of track buffers (3390 tracks, in this example) for a cache. The default cache size is one emulated cylinder (15 tracks for a 3390) for each emulated S/390 disk volume. A larger cache size may improve performance, but at the expense of using more ThinkPad memory. In general, the default cache size is acceptable.

## 3270 terminal resources

Emulated 3270 terminals, which are actually TN3270 sessions (or x3270 sessions) to a FLEX-ES server running under Linux and appear to the S/390 as local, channel-attached, non-SNA terminals, are defined as follows:

```
R1OA3174: cu 3174
  interface local(1)
  device(00) 3278 mastcon
  device(01) 3278 term701
  device(02) 3278 term702
  device(03) 3278 OFFLINE
  device(04) 3278 @9.12.17.210
  .....
  device(15) 3278 term70F
end R1OA3174
```

The interface, local, device, and device index parameters have already been described. The **3174** cutype is a provided FLEX-ES value and device type **3278** is a device type emulated by FLEX-ES. Each emulated 3270 is connected through a FLEX-ES TN3270 server. If the 3278 keyword is followed by **@IP-address** (such as @9.12.17.210 in the example), the FLEX-ES TN3270 server waits for a TN3270 connection from this IP address. If the 3278 keyword is followed by the OFFLINE keyword, then a FLEX-ES **mount** command must be issued to assign either an IP address for the connection or a name for the Terminal Solicitor. If the 3278 keyword is followed by a name (such as *mstcon* or *term70F*), this name is added to the list of terminals available through the FLEX-ES **Terminal Solicitor** (which is described in “Building a shell script” on page 29 .)

Typical mount commands, from a *flexes* prompt, might be:

```
flexes> mount 703 bills
flexes> mount 704 @9.12.17.211
```

These commands would add a new terminal name (*bills*) to the Terminal Solicitor screen, and also enable a TN3270 connection from 9.12.17.211. Where do the addresses (703 and 704) come from? The *cu* statement in our example (“Typical resource definitions” on page 76) specified an addressing range with 16 addresses. The resource name in the cu statement matches the resource set name of the 3270 definitions, so these resources have addresses beginning with 700.

The same mount commands could be placed in a shell script used to start an emulated S/390. The shell statements would be:

```
$ echo ‘mount 703 bills’ | flexescli localhost S1OA
$ echo ‘mount 704 @9.12.17.211’ | flexescli localhost S1OA
```

## Tape resources

The tape resources discussed here involve emulated S/390 tapes. If you could channel-attach “real” S/390 tapes on a ThinkPad/EFS machine, they would be defined differently, as channel devices. An emulated tape would might be a SCSI-attached device, such as a 4mm drive, a DLT drive, or an SCSI 3480/3490 unit. These are not available on ThinkPad/EFS systems, although such a system could use a FLEX-ES network channel to connect to another FLEX-ES system that has SCSI tape drives.

You can define FakeTape devices, and these would be the normal case for a ThinkPad/EFS system. The definition might be:

```
R1OA3480: cu 3480
  interface local(1)
  device(OO) 3480 /tmp/tapes/222222
end R1OA3480
```

The keyword `3480` specifies a standard FLEX-ES emulated device. The following parameter, `/tmp/tapes/222222`, is the Linux file name to be used for the emulated tape volume. FLEX-ES supports 3420, 3480, 3490, and 3490-E emulation. You should use the appropriate type to match whatever is defined in OS/390.

Several specialized parameters can be used with tape resource definitions. These would be written like this:

```
device(OO) 3480 /tmp/tapes/222222 devopt ‘maxwritesize=2OO’
```

The options are placed (in single quotes) after a *devopt* keyword. The **maxwritesize** option is probably the most commonly used. It specifies, in megabytes, the maximum size of the emulated tape media. When the amount of data written approaches this size, an end-of-tape reflective strip is emulated and the operating system would normally write EOV labels.

When using FakeTape, the tape resource would normally be specified as OFFLINE and the operator would use a flexescli **`mount`** command to specify the UNIX file to be used. (The **`mount`** command also brings the device online.) See “FLEX-ES FakeTape on OS/390” on page 45 for more information about FakeTape.

## LAN resources

LAN resources are used to define OS/390 TCP/IP connections to the S/390 or to define an SNA LAN connection to the S/390. A definition for OS/390 TCP/IP use might be:

```
R1OA3088: cu 3172
  interface local(1)
  device(OO) 3172 eth0
  device(O1) 3172 OFFLINE
end R1OA3088
```

The `cu` type `3172` is a defined FLEX-ES keyword and defines an LCS-type LAN connection to S/390. (On the S/390 side, there would normally be two CTC addresses (even/odd addresses) defined. A LAN device for use by OS/390 TCP/IP must be specified exactly as shown (except that the device name of the interface might be different). That is, it will be defined as two devices (00 and 01), with the second device OFFLINE. The IP address for this interface would be specified in the OS/390 TCP/IP profile.

The device designation in this example, `eth0`, may appear strange to experienced UNIX users, who will want to write something like /dev/eth0. However, *eth0* worked on our ThinkPad/EFS; furthermore, Linux does not appear to define names like /dev/eth0 or /dev/net0.

An additional parameter may be required for a LAN definition. A “real” 3172 unit may have up to four adapters. These are numbered 0-3, and this number appears in the OS/390 TCP/IP or VTAM specifications.[5] By default, FLEX-ES uses number zero. If you want a different number, you must include an **options** statement:

```
R1OA3088: cu 3172
  options 'adapternumber=1'
  interface local(1)
  device(00) 3172 eth0
  device(01) 3172 OFFLINE
end R1OA3088
```

This emulates the “real” 3172 adapter number 1. This is not related to the actual LAN adapter number in your ThinkPad. The actual Netfinity LAN adapter that is used is specified by the device parameter.

You can share a LAN adapter between Linux and OS/390 TCP/IPs. In order to do this, you need to specify the IP address to be used by OS/390, as a parameter in the resource definition. (You should specify the same address in your OS/390 TCP/IP parameters.) For example,

```
R1OA3088: cu 3172
  options 'ipaddress=9.12.17.211'
  interface local(1)
  device(00) 3172 eth0
  device(01) 3172 OFFLINE
end R1OA3088
```

Using this example, the single Ethernet adapter in our ThinkPad/EFS machine appears as IP address 9.12.17.210 (defined when we installed Linux) and 9.12.17.211 (when we start FLEX-ES).

## Cloned devices

The FLEX-ES documentation includes an optional **cloned** parameter for many resources definitions. We did not use this parameter. This parameter applies to a limited set of devices—emulated printers and card readers are good examples—for which the resources definitions can be automatically replicated to serve multiple S/390 instances. This is meaningful only when multiple S/390 instances are used. It potentially saves a little time when creating resources definitions. We suggest it may be better to take the extra time to directly create all the resources definitions you need and ignore the cloned parameter.

[5] The same situation exists with a P/390 or MP3000 emulated I/O LAN adapter. It is known as the *MPTS number* in these cases.

## Common rules

FLEX-ES names, in general, may be up to 255 characters and use upper/lowercase letters, numeric digits, and a few special characters. These special characters are underscore, hyphen, dollar, at, and period. A name cannot consist of a valid decimal or hexadecimal number. The following are *not* valid names: 2, 123, 1403, 0X1, 7F, 7f, 1C2a30.

White space (usually blanks) may be included almost anywhere between words. Two special cases exist:

- No white space may exist in the parameters of an *options* statement or in a *devopt* parameter. For example,

  ```
  device(00) 3390-1 /S390/VOLAs1 devopt 'trackcachesize = 10'
  ```

  is incorrect; there should not be spaces before or after the equal sign.

- If a device number range is used, there must be spaces before and after the hyphen. For example:

  ```
  device(00 - 15) 3278 OFFLINE
  ```

  is correct because there is a space before and after the hyphen.

FLEX-ES *keywords* cannot also be used as *names* in system or resources sections.

Numbers (in system and resource definitions) are always decimal. If you want a hexadecimal number, you must indicate it. For example:

```
cu devad(0xA80,10) path(1) resource(xyz)
```

contains a hexadecimal number (0xA80) and a decimal number (10). CLI *commands* differ; they assume device addresses are hexadecimal.

You should not use duplicate names in any definitions.

## The resadm command

The resadm command has a number of options that are frequently used. Using R10A.rescf as an example of a compiled resource definition, the options are:

```
# resadm -s R10A.rescf          (start the resource manager)
# resadm -k                     (kill the resource manager)
# resadm -t cu:cu3990A          (terminate an individual resource)
# resadm -T                     (terminate all resources)
$ resadm -r                     (list all active resources)
$ resadm [-h hostname] -n       (list node names)
# resadm -x R10.rescf           (refresh the resource definitions)
```

A few notes about the resource manager may be useful:

- The -r and -n options may be used by anyone. The other options are available only to *root*.
- The -r option is the most frequently used option.
- For all except the -s option, the resource manager is assumed to be running when these commands are issued.
- The -n option is for use when multiple Servers are cooperatively running FLEX-ES resources. This environment is not described in this redbook.
- You should terminate resources gracefully, with -t or -T options, instead of killing them (with -k or Linux commands).
- A common sequence is to -T (terminate all resources), then -x (refresh with a newly compiled resource file), and then re-IPL a S/390 operating system.

- You can -t (selectively terminate a resource) and then -x (refresh the resources file). This will refresh only resources that are not running at that time.
- The resource manager writes log information to the /var/log/messages file (via the Linux syslog logging facility).
- The resource manager automatically uses TCP/IP port 555 to communicate with other elements of FLEX-ES, whether there is a single Server or there are multiple Servers involved.

# CLI commands

Every S/390 emulated system instance has an associated *main console* for FLEX-ES control commands. This is a virtual console that is not directly connected to a real terminal. The commands it processes are the *CLI commands*, some of which are described here. A program named **flexescli** is used to communicate with the virtual main console. **flexescli** works from a Linux command line, and you can start as many **flexescli** instances as you like (and work via FLEX-ES networking if you have multiple FLEX-ES servers). The **flexescli** program sends commands to a *main console* and returns the results.

In discussions we typically ignore the details of the **flexescli** and *main console* interaction and describe operations in terms of commands to **flexescli**. Also, for our ThinkPad/EFS discussion, we usually assume that only a single S/390 is being emulated at any given time.

CLI commands can be entered two ways:

- You can start the **flexescli** program in interactive mode and then issue CLI commands directly at the *flexes* prompt provided by the CLI program.
- You can use ECHO to pipe a command to **flexescli**. In this case, it executes the command and terminates.

The syntax for the **flexescli** command is:

```
# flexescli IPname systemname
```

If the **flexescli** program is not in the current PATH, you would need to issue the full path name for it. The `IPname` can be an IP address, a host name that is resolved by DNS, or a local name (in /etc/hosts). The standard IPname `localhost` is used to reference the local IP loopback address. The `systemname` is the name of a syscf file used to start an emulated S/390. In our examples, this is S10A.syscf. (The `syscf` suffix is omitted for the **flexescli** command.) The **flexescli** program must always be directed to a specific S/390 instance (even if there is only one running).

An example of executing a single command through the CLI interface might be:

```
# pwd
# echo 'ipl a80 0a8200' | flexescli localhost S10A
# ...   --------------       |         |        +-- S/390 system name (S10A.syscf)
                |         |    |         +----------- local system (TCP/IP loopback)
                |         |    +--------------------- the flexescli command
                |         +-------------------------- the pipe operator
                +------------------------------------ CLI command to be executed
```

The **flexescli** program senses whether input is waiting for it in a pipe; if so, it executes the waiting command and quits. If started this way:

```
# flexescli localhost S10A
flexes>
```

it issues a flexes prompt and waits for commands. It will run until it receives a **quit** command.

There are a considerable number of CLI commands. Many of them are for low-level S/390 hands-on operation (such as instruction stepping, inspecting registers and memory, and so forth). These are described in detail in *FSIMM210: CLI Language Reference,* from FSI. We describe only a few of the higher-level commands here. We assume you are in **flexescli** interactive mode for these examples:

```
flexes> halt                                    (stops current emulated S/390 CPUs)
flexes> go                                      (resumes operation, if not single step)
flexes> hwc .....                               (enter text for system console)
flexes> ipl devaddr [parm]                      (IPLs the first emulated CPU)
flexes> iplc devaddr [parm]                     (clears state and memory first)
flexes> mount devaddr [filename | OFFLINE]      (discussed later)
flexes> notready devaddr                        (make emulated device not ready)
flexes> quit                                    (terminate flexescli; also 'exit')
flexes> ready devaddr                           (makes emulated device ready)
flexes> restart                                 (like the 'restart' button on S/390)
flexes> rewind devaddr                          (only for FakeTape or SCSI drives)
flexes> shutdown                                (gracefully terminates S/390)
flexes> unload devaddr                          (unloads FakeTape or SCSI drive)
            (like a 'mount devaddr OFFLINE' for other devices)
```

Here are a few brief notes concerning CLI commands:

- Unlike with the resources manager, CLI numeric parameters are assumed to be hexadecimal.
- There are many aliases for commands, not shown here. For example: **go** = **g** = **start**.
- The **mount** command is perhaps the most common CLI command. Do not confuse it with the Linux **mount** command. They are completely different commands that perform different functions.
  - A resources definition can define a device as OFFLINE. This means that the emulated device exists but is, in effect, turned off.
  - A CLI **mount** command can, while an emulated S/390 is running, dynamically "turn on" the device, using a specified file or name. Some examples are:

    ```
    flexes> mount A90 /s390/WORK01              (assume A90 is a 3390)
    flexes> mount 710 altcons                   (assume 710 is a local 3270)
    flexes> mount 560 /home/tape3               (a FakeTape file)
    ```

  - The first example is similar to mounting a disk pack (if disk packs were available for 3390s). The named file (or symbolic link) should point to a properly formatted emulated 3390 volume.
  - The second example causes the name *altcons* to be added to the list of connections available through the Terminal Solicitor.
  - The third example, in effect, is a tape mount.

# Special notices

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively

© Copyright IBM Corp. 2001

through The Open Group.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

# Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

## IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page 87.

- *Numa-Q Enabled for S/390: Technical Introduction,* SG24-6215
- *S/390 Partners in Development: NetFinityEnabled for S/390*, SG24-6501

### Other resources

These publications from Fundamental Software, Inc., are also relevant as further information sources:

- *FLEX-ES Concepts,* FSIMM020
- *FLEX-ES Technical FAQ,* FSIMM040
- *FLEX-ES Planning Guide,* FSIMM100
- *FLEX-ES Operator’s Guide*, FSIMM200
- *FLEX-ES CLI Language Reference*, FSIMM210
- *FLEX-ES System Programmer’s Guide*, FSIMM300
- *FLEX-ES Resource Language Reference*, FSIMM310

## Referenced Web sites

These Web sites are also relevant as further information sources:

- `http://www.funsoft.com/` Fundamental Software, Inc., home site
- `http://support.funsoft.com/` Fundamental Software, Inc., support

## How to get IBM Redbooks

Search for additional Redbooks or Redpieces, view, download, or order hardcopy from the Redbooks Web site:

**ibm.com**/redbooks

Also download additional materials (code samples or diskette/CD-ROM images) from this Redbooks site.

Redpieces are Redbooks in progress; not all Redbooks become Redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

© Copyright IBM Corp. 2001

## IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

# Index


## Symbols
/holding 42
/s391 42
/usr/flexes/bin 28, 48
/usr/flexes/rundir 28

## Numerics
1403 carriage tape 66
3174 or 3274 control units 63
3270 terminal resources 78
3339 cylinders 64
3390 drives, odd size 64
4mm tapes, compatibility 64

## A
AD releases 44
AD system 21
ADRDSSU backups 56
ASCII telnet sessions 18, 65
AWSCKD format 22
AWSOMA format 46
AWSTAPE format 46

## B
backup and restore 53
battery life 68
boot manager 63
business partners, IBM 2

## C
cachesize parameter 52, 74
cachestats command 52
CD-ROM drive 42, 43
CD-ROM formats 22
CD-ROM, mounting 24
CD-ROM, OS/390 22
CD-RW drive 56
cfcomp command 28, 29, 75
channel 74
channel emulation 63
chkckd command 58
CKD disk resources 77
ckdcachestats command 51
ckdconvaws command 23, 24, 25, 26, 38
ckdconvaws program 25
ckdfmt command 38, 42
CLI commands 82
CLI program 30
cloned parameter 80
command line interface (CLI) 18
consoles, sessions, screens 61
control unit types, emulated 75
control units and channels, which 63
cpu parameters 74
cu statement 75

## D
DASD cache 40
debugging, basic 38
dedicate Server processor(s) 17
dedicated keyword 74
defrag command 53
devad 75
device types, emulated 76
device(00) parameter 78
devopt 78
devstate command 52
DHCP 67
disk cache 41, 50
disk fragmentation 53
disk layout 44
diskette drive 42, 43
DIX frames 68
dongle 64

## E
ECC memory 57
emulated S/390 devices, maximum 64
end statement 75
ESCON and parallel channels 5
ESCON Channel Adapter 52
ESCON director 66
essize, definition 74
eth0, resource 80
Ethernet adapters, how many 64
Ethernet connection, speed 67

## F
FakeTape 65
FCBs emulated 66
fdisk 43
flexes command 29, 47
FLEX-ES console 16
FLEX-ES formats 23
FLEX-ES installation 16, 19
FLEX-ES instance 6
FLEX-ES license 3
FLEX-ES license key, installation 20
FLEX-ES manuals 65
FLEX-ES names, rules 81
FLEX-ES releases 65
FLEX-ES remote resources 66
FLEX-ES upgrades 59
flexes userid and password 19
FLEX-ES, product 3


© Copyright IBM Corp. 2001

flexescli program 82
floating point performance 67
FSI channel adapters 64
FSI hardware adapters 52
ftp 55
Fundamental Software, Incorporated 3

## H
hwc command 39

## I
ICA Adapter 52
ICA adapter 64
ICKDSF 42
IEEE802.3 frames 68
initawstape utility 46
instance, definition 6
instances, multiple 18
instruction cache 41
instset parameter 74
Integrated Server (IS) 4
interface local(1) parameter 77
IOCDS 33
IODF device definitions 23
IODF requirements 33
IP address, changing 67
ipl command 31

## K
key mapping, x3270 49

## L
LAN adapter, shared 34, 80
LAN adapters 18
LAN resources 79
license key 67
Linux commands 60
Linux swapping 41
Linux windows 58
local 75
logo, USSTAB 65

## M
MAC address 64
maxwritesize option 47
memory allocations 41
memory definition 77
memory, server 18
memsize, definition 73
MIPS 67
mount command 70, 79
mount command, Linux 24
MPTS adapter number 34
MPTS number 80
Multiprise 3000 4
Multi-system setup 47

## N
N3270 sessions, limit 65
named pipe 25

## O
Open UNIX 2
Operating Systems Messages console 39
OS/390 AD systems 21
OS/390 device configuration 23
OS/390 R10 AD system 23
OS/390 TCP/IP 18

## P
P/390, R/390 4
Parallel Channel Adapter 52
Partners in Development 2, 16, 21
PATH environment 20, 28
path parameter 75
pipe 25
port 24 18, 31
power control 59
power supplies, redundant 57
printers 68
processor speeds 65
processor, definition 6
PSW, display 57

## R
RAS characteristics 5
RAS discussion 57
raw disks 53
Red Hat 7.1 Linux 4
Redbooks Web site 87
  Contact us viii
remote resources 49, 66
resadm command 28, 29, 33, 38
resadm command, options 81
resource definitions 75
resource manager 16
Resource Measurement Facility (RMF) 33
resource parameter 75
resources definitions 28, 69
rpm commands 59

## S
S/390 central storage 64
S/390 CPU, definition 6
S/390 expanded storage 65
S/390 identification 56
SCSI adapter 53
second Linux hard disk 42
security 39
serial number, CPU 56
Server memory 39
shell script 29
shutdown process 59
shutting down 35

SNA over Ethernet 68
StarterPak systems 4
startup process 58
STIDP instruction 56
suspend function 59
swap (paging) rates 41
system console 39
system definitions 69, 73

## T

tape devices, emulated 45
tape resources 79
tar and ftp 55
tar to back up S/390 volumes 54
TCP/IP for OS/39 34
TCP/IP port 555 39, 49
terminal logo 65
Terminal Solicitor 16, 18, 31, 39, 65, 78
Terminology 6
text editors 61
ThinkPad/EFS, name 6
TN3270 client 18
TN3270 emulator, Linux 33
TN3270 server 18
TN3270 session 32
token ring LAN adapters 64
trace, processor event 66
trace, processor instruction 66
traces, I/O device 66
tracesize parameter 74
trackcachesize 51, 78
TSO and CICS users 65
TSO user 32

## U

Ultrabay 42
Uninterruptable Power Supply (UPS) 66
UNIX System Services 18
UnixWare version of FLEX-ES 64
unzip program 24, 25
unzip, wrong order 26
userids, Linux 63
USSTAB logo 31
USSTAB module 65

## V

vi editor 63
VM/ESA 6
vmstat command 40
VSE/ESA 6

## W

Windows 63
writeback 51
writeback cache 51
writethrough cache 51
writethroughcache parameter 51, 78

## X

x/Server EFS 4
x3270 client 48
x3270 command 30, 48
X3270 installation 20
x3270 sessions 18, 65

## Z

z/OS 65
z/OS status 1
zip files, separate 26
ZIPed files 22

IBM Redbooks

# S/390 Partners in Development: ThinkPad Enabled for S/390

(0.2”spine)
0.17”<->0.473”
90<->249 pages

# S/390 Partners in Development: ThinkPad Enabled for S/390

Redbooks

**System setup**

**AD CD-ROM use**

**System operation**

A ThinkPad Enabled for S/390, generally known as a ThinkPad/EFS system, is the smallest S/390 currently available, and is intended for development and demonstration purposes.

It is based on an IBM ThinkPad running Linux, and the S/390 emulation product FLEX-ES. FLEX-ES is a product of Fundamental Software, Incorporated (FSI) of Fremont, California. This package (the ThinkPad, Linux, and FLEX-ES) can run all current S/390 operating systems.

This IBM Redbook introduces the ThinkPad/EFS system, describes the setup process of the system in some detail, and then describes the installation and use of an OS/390 package known as the OS/390 AD CD-ROM system.

While this publication is primarily directed at members of the IBM S/390 Partners in Development program (also known as PartnerWorld), most of the content applies to any ThinkPad/EFS system.

**INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION**

**BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:**
**ibm.com**/redbooks

SG24-6507-00 ISBN 073842370X